DATABASE FOR COMPUTER REPAIR SHOP

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

1 PROBLEM SPECIFICATION

The owners of a computer repair shop would like to keep track of the repair jobs for computers they repair, the items used for each repair job, the labor costs for each repair job, the repairmen performing each repair job, and the total cost of each repair job.

When customers bring their computers in to be repaired, they make a deposit on the repair job and are given a date to return and uplift their computer. Repairmen then perform repairs on the customers' computers based on the repair job and detail the labor costs and the items used for each repair job.

When customers return, they pay the total cost of the repair job less the deposit, collect a receipt for their payment, and uplift the repaired computer using this payment receipt.

2 SOLUTION STEPS

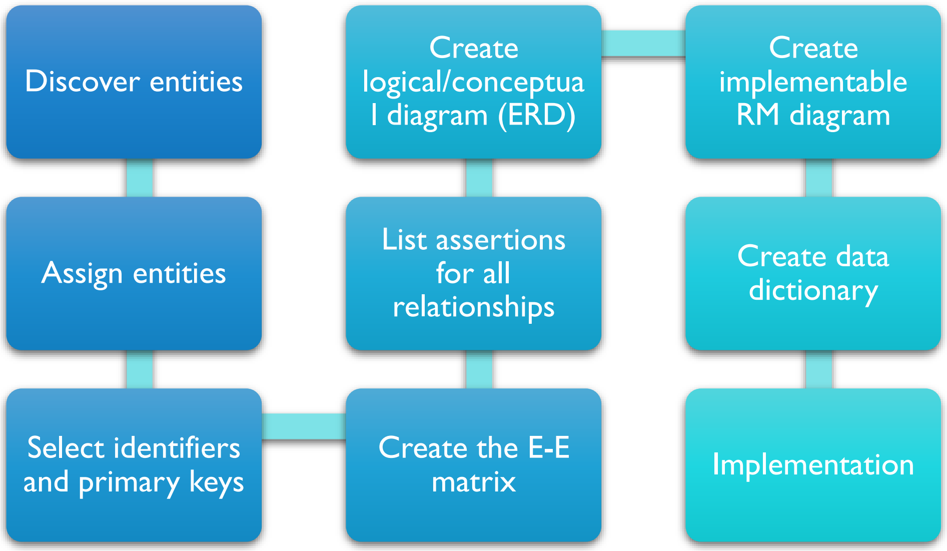

Discover entities
Assign entities
Select identifiers and primary keys
Create the E-E matrix
List assertions for all relationships
Create logical/conceptual diagram (ERD)
Create implementable RM diagram
Create data dictionary
Implementation

Figure 1: Workflow diagram.

2.1 Discover Entities

List of all the discovered entities:

- *Repair Jobs*
- *Computers*
- *Items*
- *Repairmen*
- *Customers*
- *Deposits*
- *Payment Receipts (Payments)*

2.2 Assign Attributes

list of the possible properties and/or characteristics recorded in the problem domain and those relevant to the client.

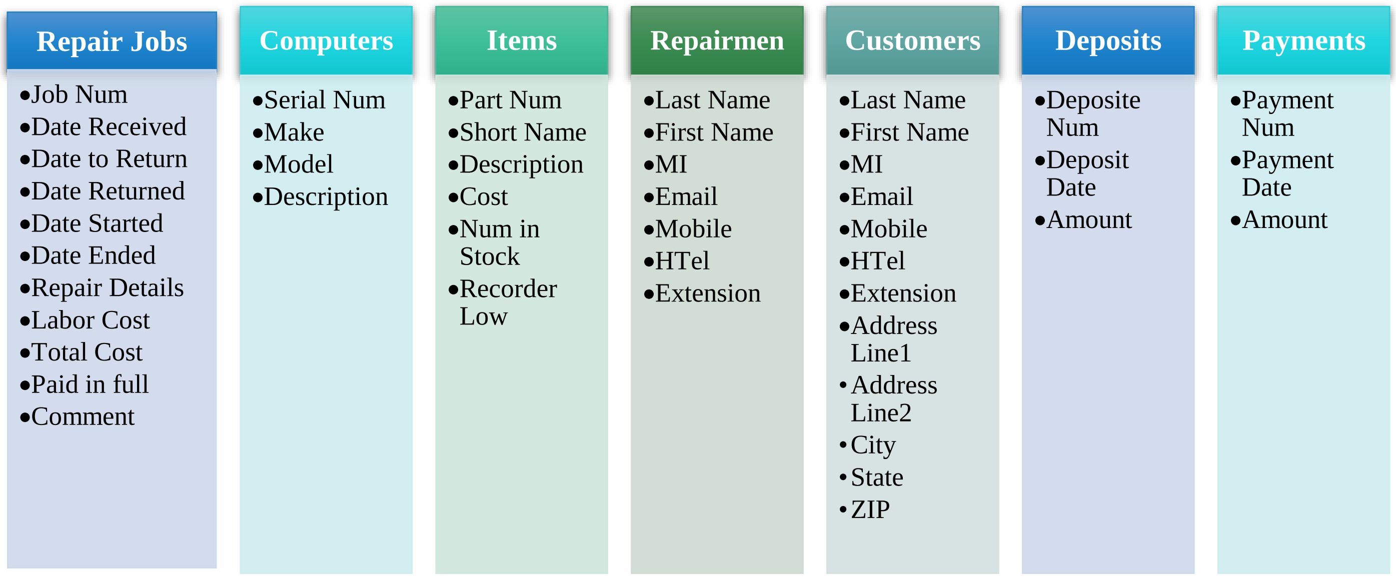


Figure 2:Entities and attributes of computer repair shop database.

2.3 Select identifiers and primary keys from attributes of each entity

Every attribute must depend on the primary key, the whole primary key.

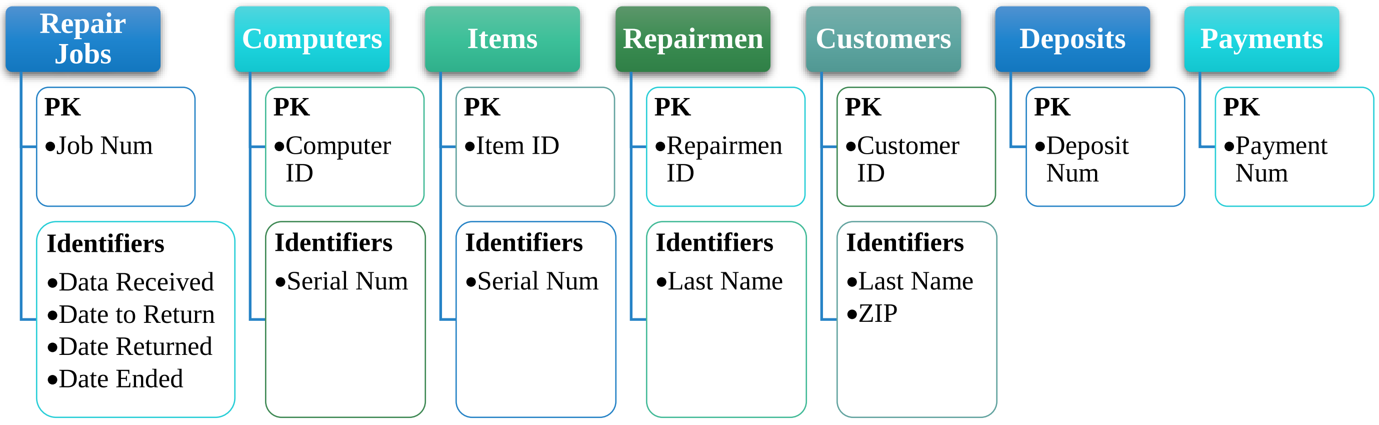


Figure 3: PK and identifiers of computer repair shop database.

2.4 Build the E-E Matrix

The intersection of the rows and columns represents the relationships that may exist between the entities

- *Diagonal* → Express the unary relationships (No unary relationship in the problem).
- *Empty cell* → No Relationships between the 2 rational entities.

Table 1: E-E Matrix of computer repair shop

	Repair Jobs	Computers	Items	Repairmen	Customers	Deposits	Payments
Repair Jobs							
Computers	**Conducted**						
Items	**Use**						
Repairmen	**Perform**		**Order**				
Customers	**Request**	**Own**					
Deposits	**Make**						
Payments	**Make**						

2.5 List assertions for all relationships (Optionality: Cardinality)

By looking at each relationship from Entity A to Entity B and writing out the relationship in words, using the entities involved in the relationship, the optionality, and cardinalities. Then, look at each relationship in reverse, from Entity B to Entity A, and write out the relationship in words, using the entities involved in the relationship, the optionality, and the cardinalities.

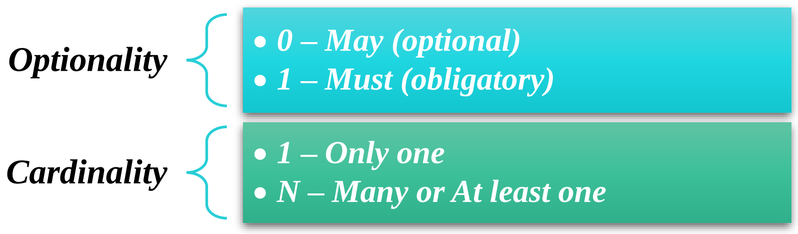


Figure 4: Assertion rules.

First, we list the assertions of the 8 relationships, mentioned in the E-E matrix above, as follows:

- A Repair Job **must** be <u>conducted</u> on **only one** Computer → (1: 1)
- A Repair Job **may** <u>use</u> **many** Items → (0: N)
- A Repair Job **must** be <u>requested</u> by **only one** Customer → (1: 1)

- A Repair Job **must** be performed by **at least one** Repairman → (1: N)
- A Repair Job **must** have made **at least one** Deposit → (1: N)
- A Repair Job **may** have made **many** Payments → (0: N)
- An Item **may** be ordered by **many** Repairmen → (0: N)
- A Customer **must** own **at least one** Computer → (1: N)

Second, we list the reverse of the 8 relationships as follows:

- Each Computer **must** have conducted **at least one** Repair Jobs → (1: N)
- Each Item **may be** used in **many** Repair Jobs → (0: N)
- Each Customer **may** request **many** Repair Jobs → (0: N)
- Each Repairman **may** perform **many** Repair Jobs → (0: N)
- Each Deposit **must** be made for **only one** Repair Job → (1: 1)
- Each Payment **must** be made for **only one** Repair Job → (1: 1)
- Each Repairman **may** order **many** Items → (0: N)
- Each Computer **must** be owned by **only one** Customer → (1: 1)

2.6 Create ER Diagram

Using the documented information above, I created this ERD using the Draw.io website.

1. With the aid of section 2.1, I created the entities as rectangles.
2. With the aid of section 2.2, I Listed the attributes of each entity.
3. With the aid of section 2.3, I distinguish the primary keys by bolding and underlining the font and the identifiers by bolding only.
4. With the aid of section 2.4, I created the relationships as diamonds.
5. Join the relationship (diamond) with its 2 entities (rectangles).
6. With the aid of section 2.5, I positioned the assertion on the 2 sides of the relationship as *(optionality: cardinality).*

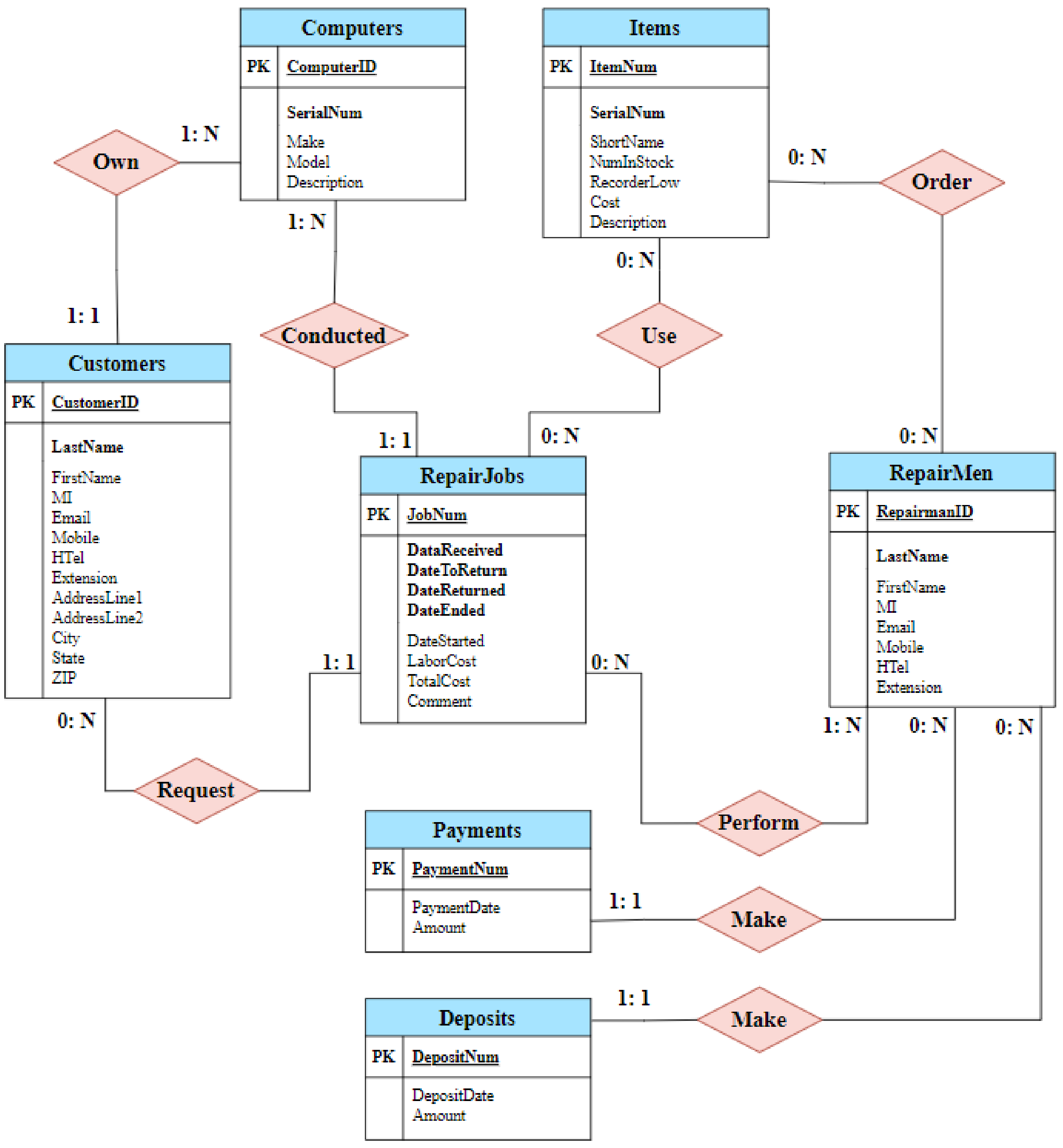


Figure 5: ERD of the computer repair shop.

2.7 Create Implementable RM Diagram

Transforming the detailed ER diagram into a Crow's Foot RM diagram and handling all many-to-many relationships. Crow's feet can only represent one-to-one and one-to-many relationships. Many-to-many relationships are created using a join-table.

To identify the many-to-many relationships, I created new (1: N), or (1: N) relationships that connect the two existing Relations to the new Relation. The many sides (Crow's Foot) of the two new relationships should be on the new Relation. Ensure that the primary key for each of the two existing Relations becomes a foreign key in the new Relation and create a new and separate primary key for the new Relation, such as a unique identifier (Id) the new Relations handled as follows:

Note: some attributes may be transported to the new Relation between the 2 sides many-to-many when it belongs to both sides.

RepairmenItems

I created a new Relation posed between Repairmen and Items Relations with 3 attributes to define more information about the item’s orders by the repairmen.

- *DateOrdered – the date at which the repairman ordered an item.*
- *Quantity – number of ordered items.*
- *TotalCost – total cost (cost of item * quantity).*

Note: TotalCost is a delivered attribute.

RepairJobsItems

I created a new Relation posed between RepairJobs and Items Relations with 3 attributes to define more information about the items used in the repair job.

- *DateUsed – the date at which the repair job used an item.*
- *Quantity – Number of used items.*
- *TotalCost – total cost (cost of item * quantity).*

RepairJobsOfRepairmen

I created a new Relation posed between RepairJobs and Repairmen Relations and transport the 3 attributes (DateStarted, DateEnded, and Comment) of the ReparJobs to this new Relation.

- *DateStarted – the date at which the repairman starts working in a repair job.*
- *DateEnded – the date at which the repairman ends the repair job.*
- *Comment.*

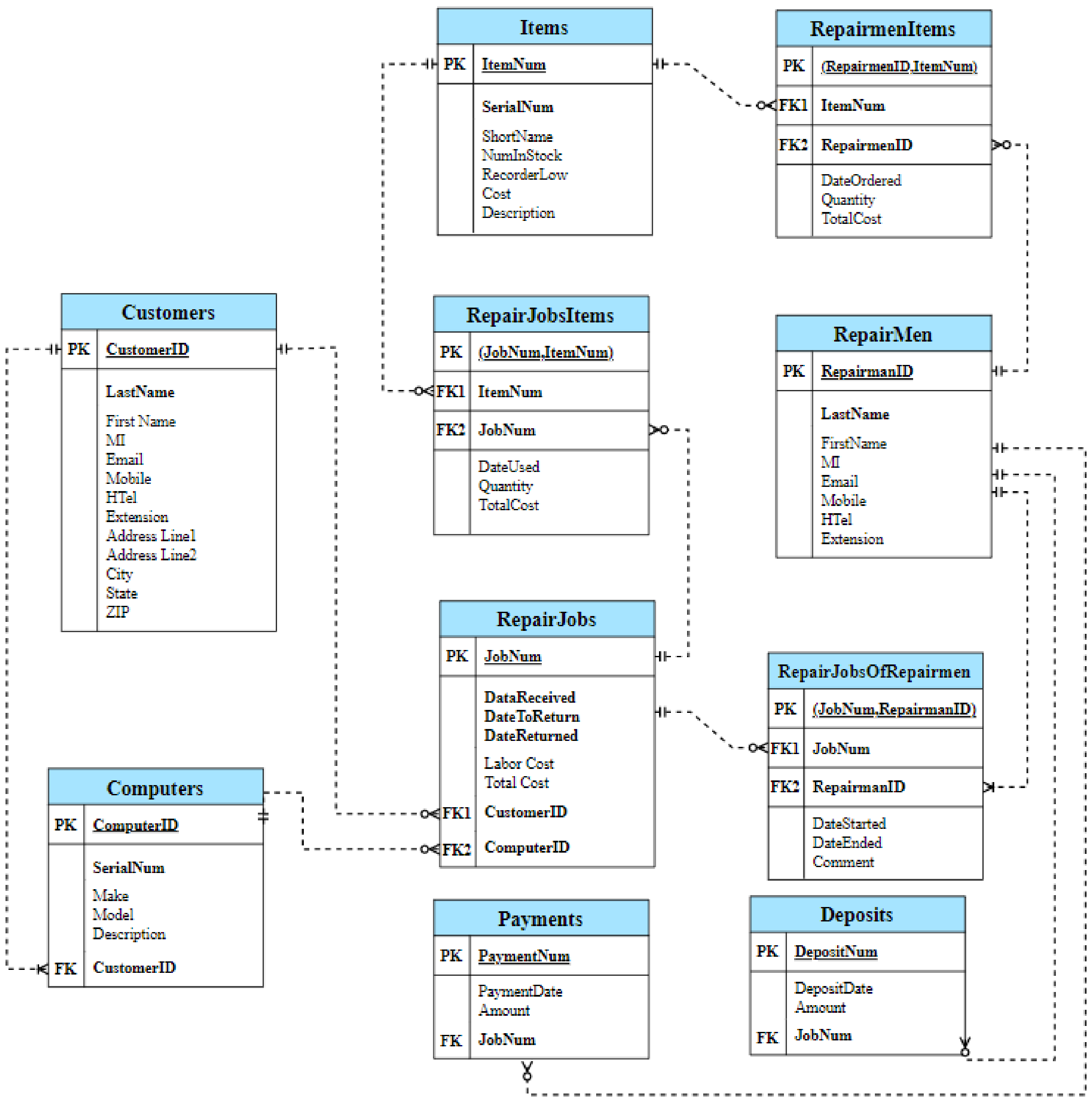


Figure 6: crow's foot RM Diagram of the computer repair shop.

2.8 Create Data Dictionary

The table below gives a breakdown of the possible data types along with the other important considerations that need to be taken into account when implementing the columns of the tables.

Table 2: Database dictionary

Table Name	Columns	Data Type	Length	Indexed	Required (Default)
Items	**ItemNum**	INT	NA	Yes, PK	Yes
	SerialNum	VARCHAR	50	Yes	Yes
	ShortName	VARCHAR	75	Yes	Yes
	Cost	DECIMAL	12	No	Yes (0.00)
	NumInStock	INT	6	No	Yes (1)
Repairmen	**RepairmenID**	INT	NA	Yes, PK	Yes
	LastName	VARCHAR	50	Yes, Composite	Yes
	FirstName	VARCHAR	50	No	Yes
	MI	CHAR	1	No	No
	Email	VARCHAR	75	No	No
	Mobile	VARCHAR	14	No	Yes
	HTel	VARCHAR	50	No	No
	Extension	VARCHAR	5	No	No
Customers	**CustomerID**	INT	NA	Yes, PK	Yes
	LastName	VARCHAR	50	Yes, Composite	Yes
	FirstName	VARCHAR	50	No	Yes
	MI	CHAR	1	No	No
	Email	VARCHAR	75	No	No
	Mobile	VARCHAR	14	No	Yes
	HTel	VARCHAR	50	No	No
	AddressLine1	VARCHAR	75	No	Yes
	AddressLine2	VARCHAR	75	No	No
	City	VARCHAR	50	No	Yes
	State	VARCHAR	50	No	Yes
	ZIP	VARCHAR	50	No	Yes

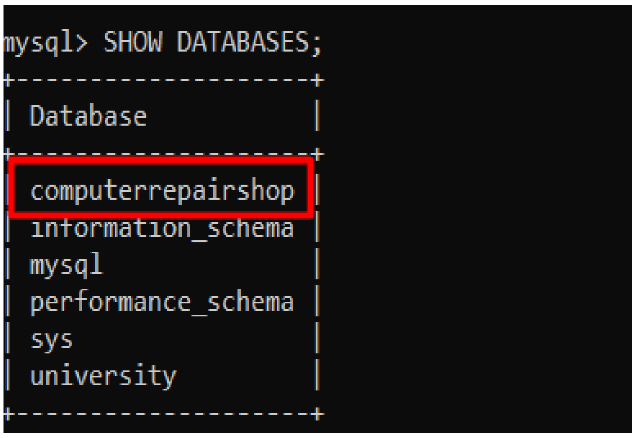
Computers	**ComputerID**	INT	NA	Yes, PK	Yes
	SerialNum	VARCHAR	50	Yes	Yes
	Make	VARCHAR	50	No	No
	Model	VARCHAR	50	No	No
	Description	VARCHAR	255	No	No
	CustomerID	INT	NA	Yes, FK	Yes
Deposits	**DepositNum**	INT	NA	Yes, PK	Yes
	DepositDate	DATE	NA	No	No
	Amount	INT	12	No	Yes (0)
	ItemNum	INT	NA	Yes, FK	Yes
Payments	**PaymentNum**	INT	NA	Yes, PK	Yes
	PaymentDate	DATE	NA	No	No
	Amount	INT	12	No	Yes (0)
	ItemNum	INT	NA	Yes, FK	Yes
RepairJobs	**JobNum**	INT	NA	Yes, PK	Yes
	DateReceived	DATE	NA	Yes	Yes
	DatetoReturn	DATE	NA	Yes	Yes
	DateReturned	DATE	NA	Yes	No
	RepairDetails	VARCHAR	255	No	No
	LaborCost	DECIMAL	12	No	Yes (0.00)
	TotalCost	DECIMAL	12	No	Yes (0.00)
	CustomerID	INT	NA	Yes, FK	Yes
	ComputerID	INT	NA	Yes, FK	Yes
RepairJob Repairmen	**(JobNum,RepairmenId)**	INT	NA	Yes, PK, Composite	Yes
	JobNum	INT	NA	Yes, FK	Yes
	RepairmenId	INT	NA	Yes, FK	Yes
	DateStarted	DATE	NA	No	Yes
	DateEnded	DATE	NA	No	Yes

	TotalCost	DECIMAL	12	No	Yes (0.00)
	Comment	TXET	NA	No	No
Repairmen Items	**(ItemNum,RepairmenId)**	INT	NA	Yes, PK, Composite	Yes
	ItemNum	INT	NA	Yes, FK	Yes
	RepairmenId	INT	NA	Yes, FK	Yes
	DateOrdered	DATE	NA	No	No
	Quantity	INT	6	No	Yes (1)
	TotalCost	DECIMAL	12	No	Yes (0.00)
RepairJob Items	**(JobNum, ItemNum)**	INT	NA	Yes, PK, Composite	Yes
	JobNum	INT	NA	Yes, FK	Yes
	ItemNum	INT	NA	Yes, FK	Yes
	DateUsed	DATE	NA	No	No
	Quantity	INT	6	No	Yes (1)
	TotalCost	DECIMAL	12	No	Yes (0.00)

2.9 Indexes

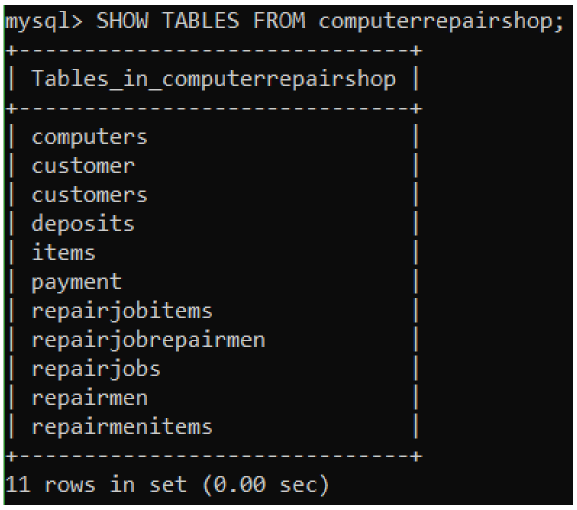
- *All mentioned indexes ON UPDATE CASCADE.*
- *All mentioned indexes ON DELETE RESTRACT.*

3 IMPLEMENTATION SNAPSHOTS



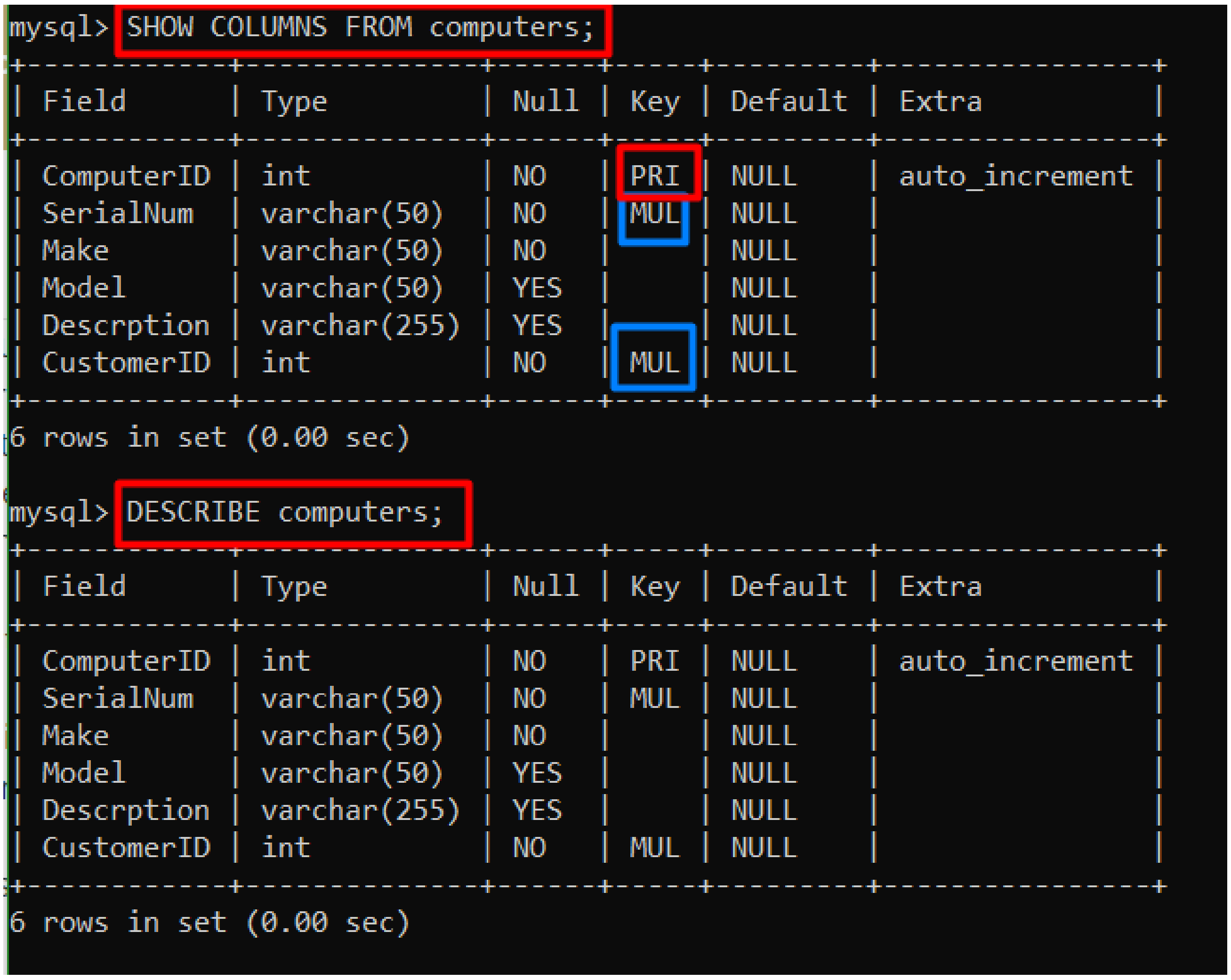
```
mysql> SHOW DATABASES;
+--------------------+
| Database           |
+--------------------+
| computerrepairshop |
| information_schema |
| mysql              |
| performance_schema |
| sys                |
| university         |
+--------------------+
```

Figure 7: Show exists databases from MySQL Command Line.



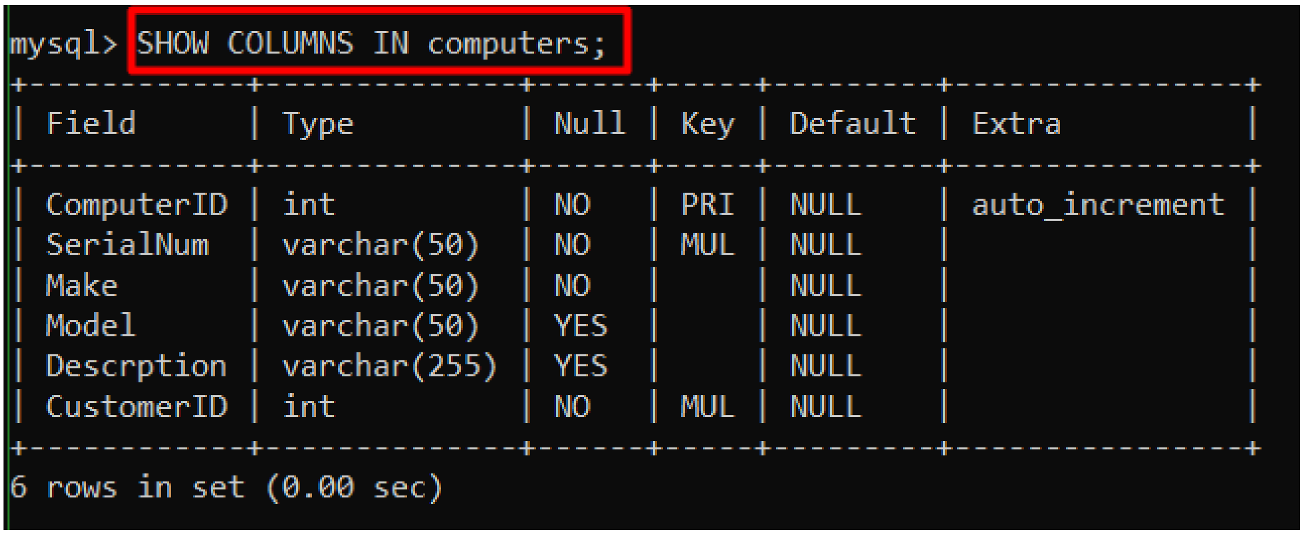
```
mysql> SHOW TABLES FROM computerrepairshop;
+------------------------------+
| Tables_in_computerrepairshop |
+------------------------------+
| computers                    |
| customer                     |
| customers                    |
| deposits                     |
| items                        |
| payment                      |
| repairjobitems               |
| repairjobrepairmen           |
| repairjobs                   |
| repairmen                    |
| repairmenitems               |
+------------------------------+
11 rows in set (0.00 sec)
```

Figure 8: Show database columns from MySQL Command Line.



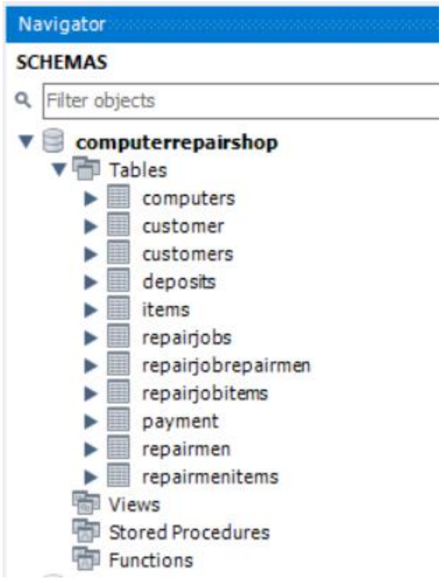
```
mysql> SHOW COLUMNS FROM computers;
+------------+--------------+------+-----+---------+----------------+
| Field      | Type         | Null | Key | Default | Extra          |
+------------+--------------+------+-----+---------+----------------+
| ComputerID | int          | NO   | PRI | NULL    | auto_increment |
| SerialNum  | varchar(50)  | NO   | MUL | NULL    |                |
| Make       | varchar(50)  | NO   |     | NULL    |                |
| Model      | varchar(50)  | YES  |     | NULL    |                |
| Descrption | varchar(255) | YES  |     | NULL    |                |
| CustomerID | int          | NO   | MUL | NULL    |                |
+------------+--------------+------+-----+---------+----------------+
6 rows in set (0.00 sec)

mysql> DESCRIBE computers;
+------------+--------------+------+-----+---------+----------------+
| Field      | Type         | Null | Key | Default | Extra          |
+------------+--------------+------+-----+---------+----------------+
| ComputerID | int          | NO   | PRI | NULL    | auto_increment |
| SerialNum  | varchar(50)  | NO   | MUL | NULL    |                |
| Make       | varchar(50)  | NO   |     | NULL    |                |
| Model      | varchar(50)  | YES  |     | NULL    |                |
| Descrption | varchar(255) | YES  |     | NULL    |                |
| CustomerID | int          | NO   | MUL | NULL    |                |
+------------+--------------+------+-----+---------+----------------+
6 rows in set (0.00 sec)
```



```
mysql> SHOW COLUMNS IN computers;
+------------+--------------+------+-----+---------+----------------+
| Field      | Type         | Null | Key | Default | Extra          |
+------------+--------------+------+-----+---------+----------------+
| ComputerID | int          | NO   | PRI | NULL    | auto_increment |
| SerialNum  | varchar(50)  | NO   | MUL | NULL    |                |
| Make       | varchar(50)  | NO   |     | NULL    |                |
| Model      | varchar(50)  | YES  |     | NULL    |                |
| Descrption | varchar(255) | YES  |     | NULL    |                |
| CustomerID | int          | NO   | MUL | NULL    |                |
+------------+--------------+------+-----+---------+----------------+
6 rows in set (0.00 sec)
```

Figure 9: Show table column from MySQL Command Line.



Navigator
SCHEMAS
Filter objects

- computerrepairshop
  - Tables
    - computers
    - customer
    - customers
    - deposits
    - items
    - repairjobs
    - repairjobrepairmen
    - repairjobitems
    - payment
    - repairmen
    - repairmenitems
  - Views
  - Stored Procedures
  - Functions

Figure 10: Schema information from MySQL workbench

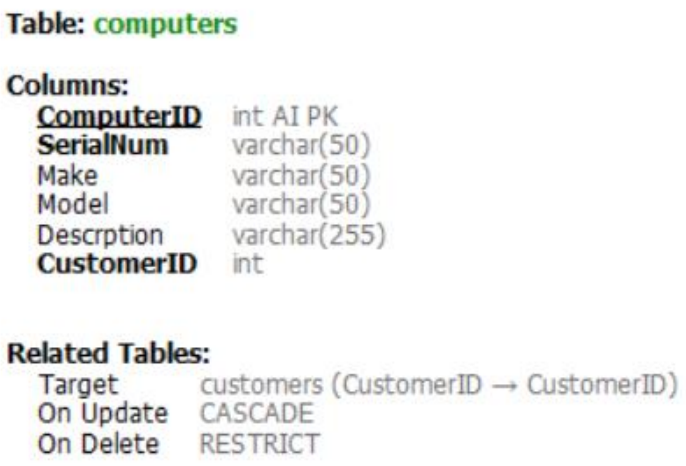


Table: computers

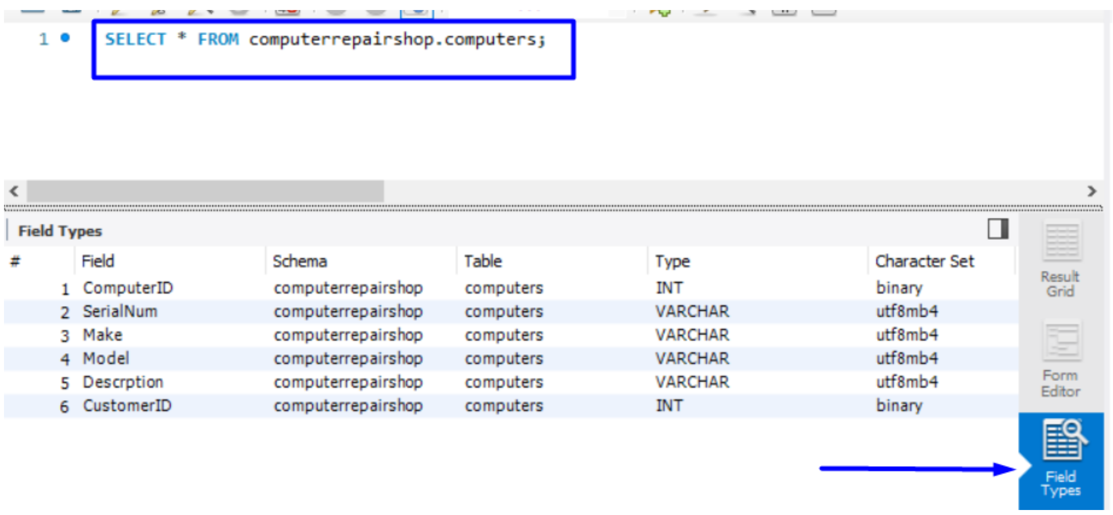
Columns:

ComputerID	int AI PK
SerialNum	varchar(50)
Make	varchar(50)
Model	varchar(50)
Descrption	varchar(255)
CustomerID	int

Related Tables:

Target	customers (CustomerID → CustomerID)
On Update	CASCADE
On Delete	RESTRICT

Figure 11: Table information from MySQL workbench.

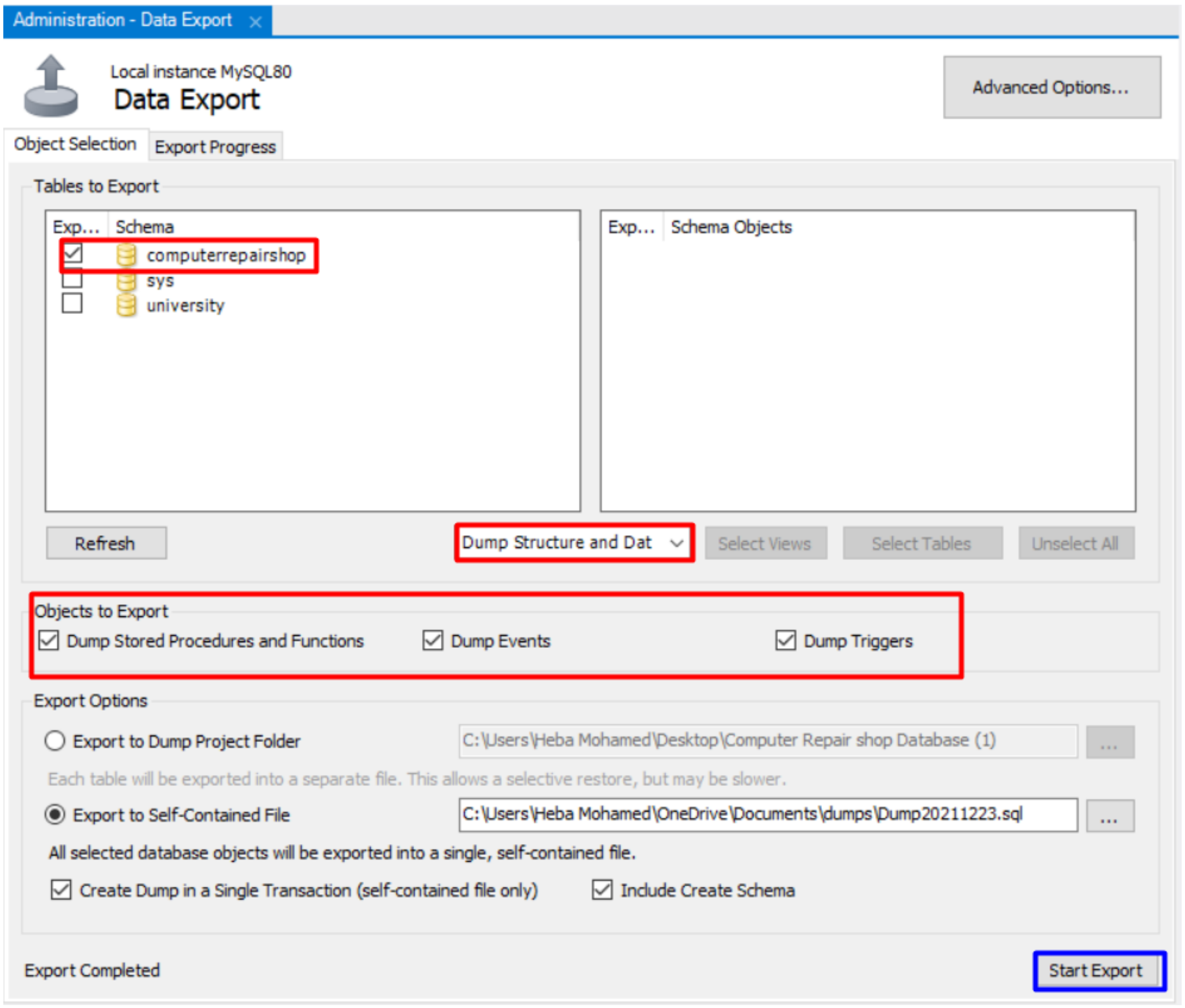


```
SELECT * FROM computerrepairshop.computers;
```

Field Types

#	Field	Schema	Table	Type	Character Set
1	ComputerID	computerrepairshop	computers	INT	binary
2	SerialNum	computerrepairshop	computers	VARCHAR	utf8mb4
3	Make	computerrepairshop	computers	VARCHAR	utf8mb4
4	Model	computerrepairshop	computers	VARCHAR	utf8mb4
5	Descrption	computerrepairshop	computers	VARCHAR	utf8mb4
6	CustomerID	computerrepairshop	computers	INT	binary

Figure 12: Show table details from MySQL workbench.



Administration - Data Export

Local instance MySQL80

Data Export

Advanced Options...

Object Selection | Export Progress

Tables to Export

Exp...	Schema
☑	computerrepairshop
☐	sys
☐	university

Exp...	Schema Objects

Refresh | Dump Structure and Dat | Select Views | Select Tables | Unselect All

Objects to Export

☑ Dump Stored Procedures and Functions ☑ Dump Events ☑ Dump Triggers

Export Options

○ Export to Dump Project Folder C:\Users\Heba Mohamed\Desktop\Computer Repair shop Database (1)

Each table will be exported into a separate file. This allows a selective restore, but may be slower.

● Export to Self-Contained File C:\Users\Heba Mohamed\OneDrive\Documents\dumps\Dump20211223.sql

All selected database objects will be exported into a single, self-contained file.

☑ Create Dump in a Single Transaction (self-contained file only) ☑ Include Create Schema

Export Completed

Start Export

Figure 13: Export the SQL Script from MySQL workbench.

4 SOFTWARE TOOLS

- *MySQL shell*
- *MySQL workbench*
- *Sublime text editor*
- *Draw.io*
- *Microsoft word*

Figure 14: Software tools.

REFERENCES

CAPTAIN, FIDEL A. *Six-Step Relational Database Design™*. Fidel A. Captain, May 2013.