Team GLASTA's *Fantastic Furniture*: Final Report

Team Info

Team Name:	Team GLASTA	
Project Name:	*Fantastic Furniture*	
Participants:	Timothy Gibson	tgibson1@csustan.edu
	Alexander Altman	aaltman@csustan.edu
	Schuyler Davis	sdavis20@csustan.edu

Contents

Team Info i

Contents **ii**

1 Part 1 **1**
1.1 Domain 1
1.2 User Group 1
1.3 Modeling Scope 1
1.4 Ground Rules 1
1.5 Possible Extensions 2

2 Part 2 **3**
2.1 Diagram 4
2.2 Explanations 5

3 Part 3 **7**
3.1 Application Domain 7
3.1.1 Domain Restrictions Not Reflected in Our Model 7
3.2 ER Diagram 8
3.3 Relational Translation 9

4 Part 4 **14**
4.1 Initial Relations 14
4.1.1 Functional Dependencies 18
4.2 Normalized Relations 19
4.2.1 Normal Forms 23

5 Part 5 **26**
5.1 Relations 26
5.2 Table Creation Statements 30
5.3 Data Counts 34
5.4 Sample Interactions 35
5.5 Data Sources 36
5.6 Data Samples 54

6 Part 6 **64**

6.1 SQL Schemas 64
6.2 Sample Queries 68

7 Part 7 **74**
7.1 Extra Functionality 74
7.2 Domain Usability 74
7.3 SQL Injection Security 74
7.4 Website Screenshots 75
Group Work 78

***Appendix*: Website Source Code and Resources** **79**
Source Code 79
`index.php` 79
`conn.php` 82
`relation.php` 82
`finder.php` 84
`query.php` 86
Resources 88
`chair.png` 88
`cabinet.png` 89
`desk.png` 89
`bedframe.png` 89
`stool.png` 90
`table.png` 90

1 Part 1

1.1 Domain

The domain of the database application will include different types of furniture. These pieces of furniture will be divided into different categories and subsets of furniture. These categories will then also have sub-categories for further narrowing of customer search results.

1.2 User Group

The intended user group for the database will be customers at a furniture store. The database will allow customers to quickly find furniture that they are shopping for. The database can be used by customers to see different types of furniture sold by the store and can sort them by different categories and characteristics such as color, size, type, and model, among others.

1.3 Modeling Scope

The system will model a database of furniture for a furniture website. The system will model different types of furniture, their categories, their different characteristics and the different applications of the furniture pieces. The system will *not* model an application that will be used by the store staff. The system will *not* include information that is important for employees who work at the store such as shipment dates, shipping routes for each piece, in stock dates, out of stock dates, or other such data. The intent of the database is to catalog all the information a customer seeking to buy furniture would find relevant.

1.4 Ground Rules

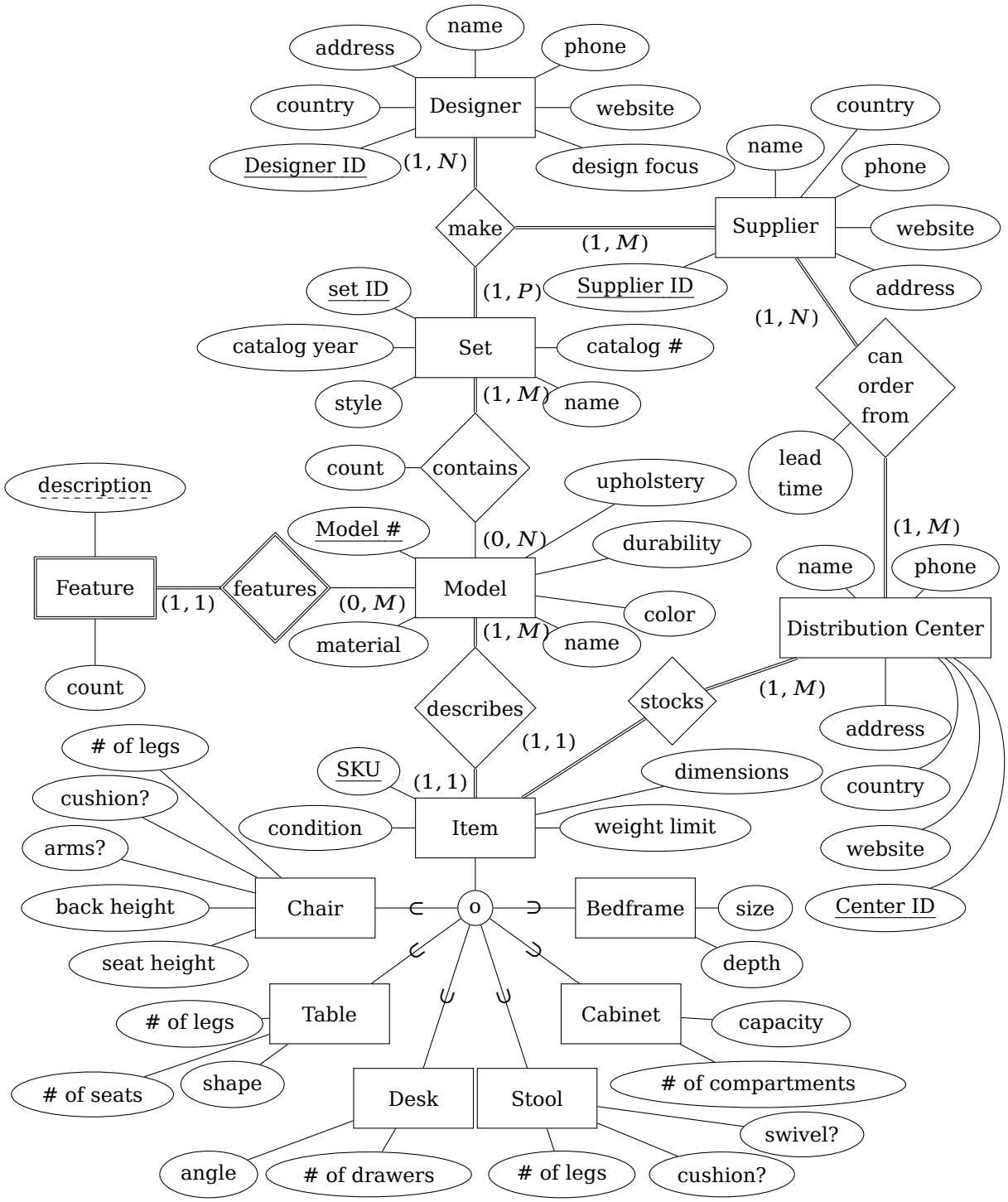
1. Complete every task assigned to you.
2. Communicate with group members about problems or concerns.
3. Give as much feedback as possible.
4. Attend every group meeting.
5. Tell group members about inability to perform tasks or attend group meetings *beforehand*.

1.5 Possible Extensions

The system could support a suggestion tool to suggest pieces of furniture to customers based on viewing or buying habits. The system could also support a furniture listing system for customers to save lists of specific pieces of furniture for future viewing or purchase. In addition, the system could enable the ability for users to search for items in the database based on different search terms, such as color, type, or SKU number, among others.

2 Part 2

2.1 Diagram


name
address
phone
country
Designer
website
Designer ID
(1, N)
design focus
country
name
phone
make
Supplier
website
(1, M)
set ID
(1, P)
Supplier ID
address
(1, N)
catalog year
Set
catalog #
can order from
style
(1, M)
name
lead time
count
contains
description
upholstery
Model #
(0, N)
durability
(1, M)
Feature
features
Model
name
phone
(1, 1)
(0, M)
color
Distribution Center
material
(1, M)
name
count
describes
stocks
(1, M)
address
# of legs
(1, 1)
SKU
(1, 1)
dimensions
country
cushion?
condition
Item
weight limit
arms?
website
back height
Chair
o
Bedframe
size
Center ID
seat height
depth
Table
Cabinet
# of legs
capacity
# of seats
shape
Desk
Stool
# of compartments
swivel?
angle
# of drawers
# of legs
cushion?

2.2 Explanations

Supplier: A person or company that supplies sets of furniture.

Designer: A person or company that designs sets of furniture.

make: What a designer and supplier do together, which is to design and then supply a set of furniture.

Set: A collection of pieces of furniture that fit into one aesthetic style or are part of a matching group of pieces.

contains: Each set of furniture incorporates many different models of furniture. These models will all fit the same color, design, or aesthetic style.

Model: The make and style of one of the pieces of furniture in a set. This may also include color, wood type, or stain of the wood.

features: Each model in a set has the option to include extra features. These features may be things such as different styled knobs or legs.

Feature: Each model in a set can include an optional item that is not normally included. These are usually extra adornments.

is finished with: Each model can be finished in a variety of textures, colors, and materials, or can be unfinished.

Finish: Each model can be chosen to have a specific finish. The finish varies by color and material and describes the aesthetic of the set.

describes: Each model describes some of the items in the store database. The items described by a given model are all "copies" of each other.

Item: The specific piece of furniture that the database lists. Items are distinguished from one another by their specific SKU number.

- **Chair:** A simple seat for one person; has three or more legs. The chair will also contain a back that the customer can sit against that is a specific height.
- **Table:** A flat surface elevated by a three or more legs. The table is distinguished by its size and shape.
- **Desk:** A flat or sloped elevated surface atop a chest of drawers or compartments.
- **Stool:** A seat without a back or arms. Must have three or more legs and no back.
- **Cabinet:** A piece of furniture used for storage, usually containing drawers or shelves. The cabinet can either sit on the floor atop a few legs or it can be mounted directly to the wall.

Bedframe: The frame of a bed that supports the mattress. This includes a headboard, a footboard, and some rails.

Distribution Center: A factory or warehouse from which items are directly shipped to the store.

can order from: A distribution center can order models, sets of models, or specific items from a particular supplier, and will do so both on a regular schedule and in response to a special request from the store.

stocks: Distribution centers will hold items until they are shipped out to the store. Items can also be directly requested from a distribution center by the store's customers.

3 Part 3

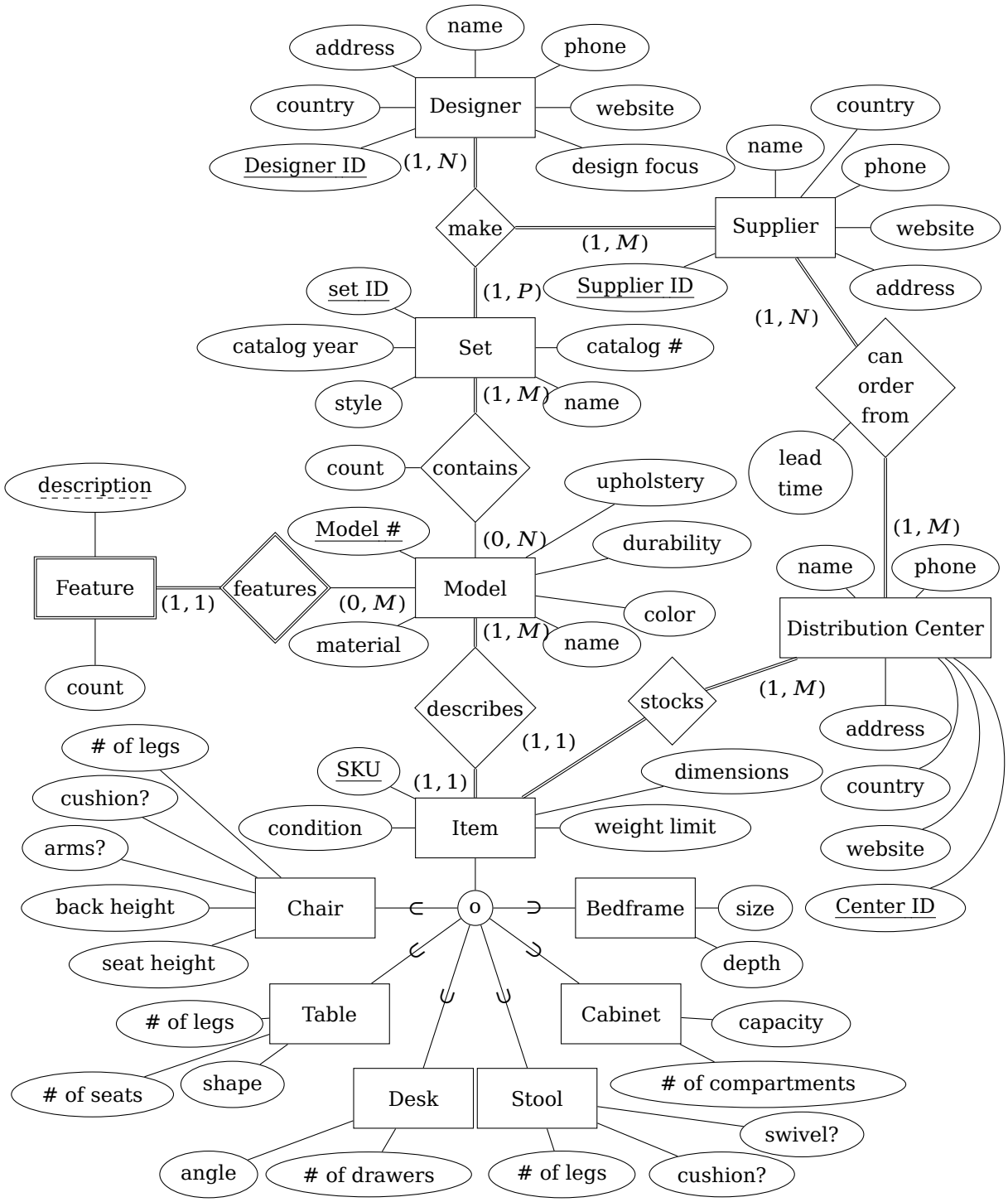
3.1 Application Domain

We will be modeling the designer and supplier of a particular piece of furniture. We will also be modeling the distribution centers that will allow customers to order furniture and have it shipped to the store. Pieces of furniture are organized into sets that have different styles. Each set contains the models of the furniture, which are part of a specific catalog and has a unique id number to reference that catalog. We are also modeling the individual items in our store which includes the physical condition on used items. Item can be broken up into an incomplete specialization hierarchy that contains: tables, chairs, bedframes, desks, stools, and cabinets.

3.1.1 Domain Restrictions Not Reflected in Our Model

We won't be covering things like cushions or mattresses in our model, nor will we be modeling the actual shipping of the products to their locations. We will also not be modeling the locations of multiple stores, but we do have the distribution network to model that one particular store.

3.2 ER Diagram



3.3 Relational Translation

Note that some of the table and column names have underscores after them; this is because those names would otherwise conflict with SQL keywords. Additionally, ISO standard SQL doesn't have any syntax for comments, so we have used the `--` syntax common in practice for this purpose.

```sql
create domain posreal as double precision
check         (value > 0.0);

create domain posint as integer
check         (value > 0);

-- all measures in this type are in inches
create type dimensions as (length_ posreal,
                           width   posreal,
                           height  posreal);

create table Supplier(supplierID  varchar(10),
                      name_       nchar varying(50)
                                  not null,
                      phone       varchar(12),
                      address     nchar varying(100),
                      country     char(2),
                      website     nchar varying(50),
                      primary key (supplierID));

create table Designer(designerID  varchar(10),
                      name_       nchar varying(50)
                                  not null,
                      phone       varchar(12),
                      address     nchar varying(100),
                      country     char(2),
                      website     nchar varying(50),
                      designFocus nchar varying(100),
                      primary key (designerID));

create table Set_(setID         varchar(10),
                  name_         nchar varying(50)
                                not null,
                  catalogYear   numeric(4,0),
                  catalogNumber integer
                                not null,
                  style_        nchar varying(30),
                  primary key   (setID));
```

```
create table Model(modelNumber varchar(10),
                   name_       nchar varying(50)
                               not null,
                   material    nchar varying(30),
                   upholstery  nchar varying(30),
                   durability  nchar varying(30),
                   color       nchar varying(30),
                   primary key (modelNumber));

create table Item(sku         varchar(10),
                  dimensions  dimensions
                              not null, -- anti-redundancy constraint
                  condition   nchar varying(30),
                  weightLimit posreal, -- in pounds of weight
                  primary key (sku));

create table DistributionCenter(centerID    varchar(10),
                                name_       nchar varying(50)
                                            not null,
                                phone       varchar(12),
                                address     nchar varying(100),
                                country     char(2),
                                website     nchar varying(50),
                                primary key (centerID));

create table make(supplierID  varchar(10),
                  designerID  varchar(10),
                  setID       varchar(10),
                  primary key (supplierID,
                               designerID,
                               setID),
                  foreign key (supplierID)
                              references Supplier,
                  foreign key (designerID)
                              references Designer,
                  foreign key (setID)
                              references Set_);

create table contains_(setID       varchar(10),
                       modelNumber varchar(10),
                       count_      posint,
                       primary key (setID,
                                    modelNumber),
                       foreign key (setID)
```

```
                                    references Set_,
                        foreign key (modelNumber)
                                    references Model);

create table describes(modelNumber varchar(10)
                                    not null,
                        sku         varchar(10),
                        primary key (sku),
                        foreign key (modelNumber)
                                    references Model,
                        foreign key (sku)
                                    references Item);

create table canOrderFrom(centerID    varchar(10),
                          supplierID  varchar(10),
                          leadTime    double precision, -- in days
                          primary key (centerID,
                                       supplierID),
                          foreign key (centerID)
                                      references DistributionCenter,
                          foreign key (supplierID)
                                      references Supplier,
                          check       (leadTime >= 0.0));

create table stocks(centerID    varchar(10)
                                not null,
                    sku         varchar(10),
                    primary key (sku),
                    foreign key (centerID)
                                references DistributionCenter,
                    foreign key (sku)
                                references Item);

create table Chair(sku          varchar(10),
                   numberOfLegs posint,
                   hasCushion   boolean,
                   hasArms      boolean,
                   backHeight   posreal, -- in inches
                   seatHeight   posreal, -- in inches
                   primary key  (sku),
                   foreign key  (sku)
                                references Item);

create table Table_(sku           varchar(10),
                    numberOfLegs  posint,
```

```
                    numberOfSeats posint,
                    shape         nchar varying(30),
                    primary key   (sku),
                    foreign key   (sku)
                                  references Item);

create table Desk(sku             varchar(10),
                  angle           double precision,
                   ↪  -- in degrees, possibly negative
                  numberOfDrawers posint,
                  primary key     (sku),
                  foreign key     (sku)
                                  references Item,
                  check              (angle > -360.0
                                  and angle < 360.0));

create table Stool(sku          varchar(10),
                   numberOfLegs posint,
                   hasCushion   boolean,
                   hasSwivel    boolean,
                   primary key  (sku),
                   foreign key  (sku)
                                references Item);

create table Cabinet(sku                  varchar(10),
                     numberOfCompartments posint,
                     capacity             nchar varying(30),
                     primary key          (sku),
                     foreign key          (sku)
                                          references Item);

create table Bedframe(sku         varchar(10),
                      size_       nchar varying(30),
                      depth_      double precision,
                       ↪  -- in inches, possibly negative
                      primary key (sku),
                      foreign key (sku)
                                  references Item);

create table features_Feature(modelNumber varchar(10),
                              description nchar varying(50),
                              count_      posint,
                              primary key (modelNumber,
                                           description),
                              foreign key (modelNumber)
```

```
172                                         references Model);
```

4 Part 4

4.1 Initial Relations

```sql
create domain posreal as double precision
check         (value > 0.0);

create domain posint as integer
check         (value > 0);

-- all measures in this type are in inches
create type dimensions as (length_ posreal,
                           width   posreal,
                           height  posreal);

create table Supplier(supplierID  varchar(10),
                      name_       nchar varying(50)
                                  not null,
                      phone       varchar(12),
                      address     nchar varying(100),
                      country     char(2),
                      website     nchar varying(50),
                      primary key (supplierID));

create table Designer(designerID  varchar(10),
                      name_       nchar varying(50)
                                  not null,
                      phone       varchar(12),
                      address     nchar varying(100),
                      country     char(2),
                      website     nchar varying(50),
                      designFocus nchar varying(100),
                      primary key (designerID));

create table Set_(setID         varchar(10),
                  name_         nchar varying(50)
```

```
                                not null,
                  catalogYear   numeric(4,0),
                  catalogNumber integer
                                not null,
                  style_        nchar varying(30),
                  primary key   (setID));

create table Model(modelNumber varchar(10),
                   name_       nchar varying(50)
                               not null,
                   material    nchar varying(30),
                   upholstery  nchar varying(30),
                   durability  nchar varying(30),
                   color       nchar varying(30),
                   primary key (modelNumber));

create table Item(sku         varchar(10),
                  dimensions  dimensions
                              not null, -- anti-redundancy constraint
                  condition   nchar varying(30),
                  weightLimit posreal, -- in pounds of weight
                  primary key (sku));

create table DistributionCenter(centerID    varchar(10),
                                name_       nchar varying(50)
                                            not null,
                                phone       varchar(12),
                                address     nchar varying(100),
                                country     char(2),
                                website     nchar varying(50),
                                primary key (centerID));

create table make(supplierID  varchar(10),
                  designerID  varchar(10),
                  setID       varchar(10),
                  primary key (supplierID,
                               designerID,
                               setID),
                  foreign key (supplierID)
                              references Supplier,
                  foreign key (designerID)
                              references Designer,
                  foreign key (setID)
                              references Set_);
```

```
create table contains_(setID       varchar(10),
                       modelNumber varchar(10),
                       count_      posint,
                       primary key (setID,
                                    modelNumber),
                       foreign key (setID)
                                   references Set_,
                       foreign key (modelNumber)
                                   references Model);

create table describes(modelNumber varchar(10)
                                   not null,
                       sku         varchar(10),
                       primary key (sku),
                       foreign key (modelNumber)
                                   references Model,
                       foreign key (sku)
                                   references Item);

create table canOrderFrom(centerID    varchar(10),
                          supplierID  varchar(10),
                          leadTime    double precision, -- in days
                          primary key (centerID,
                                       supplierID),
                          foreign key (centerID)
                                      references DistributionCenter,
                          foreign key (supplierID)
                                      references Supplier,
                          check       (leadTime >= 0.0));

create table stocks(centerID    varchar(10)
                                not null,
                    sku         varchar(10),
                    primary key (sku),
                    foreign key (centerID)
                                references DistributionCenter,
                    foreign key (sku)
                                references Item);

create table Chair(sku          varchar(10),
                   numberOfLegs posint,
                   hasCushion   boolean,
                   hasArms      boolean,
                   backHeight   posreal, -- in inches
                   seatHeight   posreal, -- in inches
```

```
                    primary key  (sku),
                    foreign key  (sku)
                                 references Item);

create table Table_(sku           varchar(10),
                     numberOfLegs  posint,
                     numberOfSeats posint,
                     shape         nchar varying(30),
                     primary key   (sku),
                     foreign key   (sku)
                                   references Item);

create table Desk(sku             varchar(10),
                  angle           double precision,
                   ↪  -- in degrees, possibly negative
                  numberOfDrawers posint,
                  primary key     (sku),
                  foreign key     (sku)
                                  references Item,
                  check              (angle > -360.0
                                  and angle < 360.0));

create table Stool(sku          varchar(10),
                    numberOfLegs posint,
                    hasCushion   boolean,
                    hasSwivel    boolean,
                    primary key  (sku),
                    foreign key  (sku)
                                 references Item);

create table Cabinet(sku                  varchar(10),
                      numberOfCompartments posint,
                      capacity             nchar varying(30),
                      primary key          (sku),
                      foreign key          (sku)
                                           references Item);

create table Bedframe(sku         varchar(10),
                       size_       nchar varying(30),
                       depth_      double precision,
                        ↪  -- in inches, possibly negative
                       primary key (sku),
                       foreign key (sku)
                                   references Item);
```

```
create table features_Feature(modelNumber varchar(10),
                              description nchar varying(50),
                              count_      posint,
                              primary key (modelNumber,
                                           description),
                              foreign key (modelNumber)
                                          references Model);
```

4.1.1 Functional Dependencies

supplierID	⟶	name_	Supplier
supplierID	⟶	phone	
supplierID	⟶	address	
supplierID	⟶	country	
supplierID	⟶	website	
designerID	⟶	name_	Designer
designerID	⟶	phone	
designerID	⟶	address	
designerID	⟶	country	
designerID	⟶	website	
designerID	⟶	designFocus	
setID	⟶	name_	Set_
setID	⟶	catalogYear	
setID	⟶	catalogNumber	
setID	⟶	style_	
modelNumber	⟶	name_	Model
modelNumber	⟶	material	
modelNumber	⟶	upholstery	
modelNumber	⟶	durability	
modelNumber	⟶	color	
sku	⟶	name_	Item
sku	⟶	dimensions.length_	
sku	⟶	dimensions.width	
sku	⟶	dimensions.height	
sku	⟶	condition	
sku	⟶	weightLimit	
centerID	⟶	name_	DistributionCenter
centerID	⟶	phone	
centerID	⟶	address	
centerID	⟶	country	
centerID	⟶	website	

Note: the `make` *relation has no nontrivial functional dependencies.*

{setID, modelNumber}	⟶	count_	contains_
sku	⟶	modelNumber	describes

{centerID, supplierID}	⟶	leadTime	canOrderFrom
sku	⟶	centerID	stocks
sku	⟶	numberOfLegs	Chair
sku	⟶	hasCushion	
sku	⟶	hasArms	
sku	⟶	backHeight	
sku	⟶	seatHeight	
sku	⟶	numberOfLegs	Table_
sku	⟶	numberOfSeats	
sku	⟶	shape	
sku	⟶	angle	Desk
sku	⟶	numberOfDrawers	
sku	⟶	numberOfLegs	Stool
sku	⟶	hasCushion	
sku	⟶	hasSwivel	
sku	⟶	numberOfCompartments	Cabinet
sku	⟶	capacity	
sku	⟶	size_	Bedframe
sku	⟶	depth_	
{modelNumber, description}	⟶	count_	features_Feature

Functional Dependency Notes

One might think, at first, that several attributes of `Supplier` (such as `website` and `phone`) should be candidate keys by virtue of uniquely determining the primary key `supplierID`. However, consider the following scenario: a supplier is an independent carpenter living in a country whose laws (for whatever reason) disallow any one single business from selling both chairs and bedframes. This carpenter produces both types of items, but, because of the laws of his home country, he has to run two separate businesses from the legal perspective. The inevitable result of us getting furniture from this carpenter is two `Supplier` tuples with (necessarily) distinct `supplierID`s but where *every other attribute is identical*! Therefore, by constructed counterexample, `Supplier`'s only candidate key is its primary key `supplierID`; very similar reasoning applies to `DistributionCenter` and `Designer` as well.

4.2 Normalized Relations

```
create domain posreal as double precision
check         (value > 0.0);

create domain posint as integer
check         (value > 0);

create table Supplier(supplierID  varchar(10),
                      name_       nchar varying(50)
                                  not null,
```

```
                     phone       varchar(12),
                     address     nchar varying(100),
                     country     char(2),
                     website     nchar varying(50),
                     primary key (supplierID));

create table Designer(designerID  varchar(10),
                     name_       nchar varying(50)
                                 not null,
                     phone       varchar(12),
                     address     nchar varying(100),
                     country     char(2),
                     website     nchar varying(50),
                     designFocus nchar varying(100),
                     primary key (designerID));

create table Set_(setID         varchar(10),
                  name_         nchar varying(50)
                                not null,
                  catalogYear   numeric(4,0),
                  catalogNumber integer
                                not null,
                  style_        nchar varying(30),
                  primary key   (setID));

create table Model(modelNumber varchar(10),
                   name_       nchar varying(50)
                               not null,
                   material    nchar varying(30),
                   upholstery  nchar varying(30),
                   durability  nchar varying(30),
                   color       nchar varying(30),
                   primary key (modelNumber));

create table Item(sku         varchar(10),
                  length_     posreal, -- in inches
                  width       posreal, -- in inches
                  height      posreal, -- in inches
                  condition   nchar varying(30),
                  weightLimit posreal, -- in pounds of weight
                  primary key (sku));

create table DistributionCenter(centerID    varchar(10),
                                name_       nchar varying(50)
                                            not null,
```

```
                                phone       varchar(12),
                                address     nchar varying(100),
                                country     char(2),
                                website     nchar varying(50),
                                primary key (centerID));

create table make(supplierID  varchar(10),
                  designerID  varchar(10),
                  setID       varchar(10),
                  primary key (supplierID,
                               designerID,
                               setID),
                  foreign key (supplierID)
                              references Supplier,
                  foreign key (designerID)
                              references Designer,
                  foreign key (setID)
                              references Set_);

create table contains_(setID       varchar(10),
                       modelNumber varchar(10),
                       count_      posint,
                       primary key (setID,
                                    modelNumber),
                       foreign key (setID)
                                   references Set_,
                       foreign key (modelNumber)
                                   references Model);

create table describes(modelNumber varchar(10)
                                   not null,
                       sku         varchar(10),
                       primary key (sku),
                       foreign key (modelNumber)
                                   references Model,
                       foreign key (sku)
                                   references Item);

create table canOrderFrom(centerID    varchar(10),
                          supplierID  varchar(10),
                          leadTime    double precision, -- in days
                          primary key (centerID,
                                       supplierID),
                          foreign key (centerID)
                                      references DistributionCenter,
```

```
                        foreign key (supplierID)
                                    references Supplier,
                        check       (leadTime >= 0.0));

create table stocks(centerID    varchar(10)
                                not null,
                    sku         varchar(10),
                    primary key (sku),
                    foreign key (centerID)
                                references DistributionCenter,
                    foreign key (sku)
                                references Item);

create table Chair(sku          varchar(10),
                   numberOfLegs posint,
                   hasCushion   boolean,
                   hasArms      boolean,
                   backHeight   posreal, -- in inches
                   seatHeight   posreal, -- in inches
                   primary key  (sku),
                   foreign key  (sku)
                                references Item);

create table Table_(sku           varchar(10),
                    numberOfLegs  posint,
                    numberOfSeats posint,
                    shape         nchar varying(30),
                    primary key   (sku),
                    foreign key   (sku)
                                  references Item);

create table Desk(sku             varchar(10),
                  angle           double precision,
                   ↪  -- in degrees, possibly negative
                  numberOfDrawers posint,
                  primary key     (sku),
                  foreign key     (sku)
                                  references Item,
                  check              (angle > -360.0
                                  and angle < 360.0));

create table Stool(sku          varchar(10),
                   numberOfLegs posint,
                   hasCushion   boolean,
                   hasSwivel    boolean,
```

```
                    primary key  (sku),
                    foreign key  (sku)
                                 references Item);

create table Cabinet(sku                  varchar(10),
                     numberOfCompartments posint,
                     capacity             nchar varying(30),
                     primary key          (sku),
                     foreign key          (sku)
                                          references Item);

create table Bedframe(sku         varchar(10),
                      size_       nchar varying(30),
                      depth_      double precision,
                       ↪  -- in inches, possibly negative
                      primary key (sku),
                      foreign key (sku)
                                  references Item);

create table features_Feature(modelNumber varchar(10),
                              description nchar varying(50),
                              count_      posint,
                              primary key (modelNumber,
                                           description),
                              foreign key (modelNumber)
                                          references Model);
```

4.2.1 Normal Forms

`Supplier` `Supplier` is in BCNF. Each functional dependency in the relation points back to `supplierID`. Every other attribute of the relation is dependent on `supplierID` and as such is a member of the candidate key. Also, every attribute of the `Supplier` relation is functionally determined by `supplierID`, making it the superkey.

`Designer` `Designer` is in BCNF. `designerID` determines every attribute of the `Designer` relation. This effectively makes `designerID` the superkey for the `Designer` relation.

`Set_` The `Set_` relation is in BCNF. `Set_` has a candidate key that is `setID`. Each attribute of `Set_` is unique to a specific `setID`. This indicates that each attribute is determined by that setID. This makes setID the superkey for `Set_`. From this, it lends that `Set_` is in BCNF because that for every non-trivial functional dependency, `setID` is the superkey.

Model Model is in BCNF. For every attribute of the Model relation, they are functionally determined by the modelNumber, meaning that they are only determined by one specific modelNumber. This makes modelNumber the superkey for Model.

Item The Item relation is in BCNF. The attribute sku is a single identifier for each individual item. Each item has one name, length_, width, height, and weightLimit. Each of these attributes are dependent on one sku. This makes sku the superkey for Item. Since all attributes are dependent on a single sku then Item is in BCNF.

In our original SQL, the Item relation used a type (dimensions) that was created specifically for that relation in its table. That type's fields have now been defined inline inside of the table so that it complies with the requirements of BCNF.

DistributionCenter DistributionCenter is in BCNF. Each name_, phone, address, country, and website is specific to one centerID. This makes centerID the superkey for the DistributionCenter relation. Since each attribute is only populated by one tuple and the superkey determines every attribute, then the DistributionCenter relation is in BCNF.

make The make relation is in BCNF, since it only contains trivial functional dependencies.

contains_ The contains_ relation is in BCNF. The contains_ relation has a primary key of {setID, modelNumber}. These two together effectively become the superkey and since the count of the contains_ relation can be determined by the setID and modelNumber, contains_ is in BCNF.

describes The describes relation is in BCNF. The describes relation includes two foreign keys that together also form the primary key. These two keys are modelNumber and sku. The modelNumber specifically determines a single sku. This means that sku is dependent on the modelNumber and that modelNumber is the superkey. Since the only non-trivial functional dependency in the describes relation involves the superkey determining the single other attribute, describes is in BCNF.

canOrderFrom The canOrderFrom relation is in BCNF. The canOrderFrom relation has an attribute called leadTime. This leadTime is dependent on the distribution center (identified by centerID) and the supplier (identified by supplierID). This centerID and supplierID are both part of the primary key for canOrderFrom and together make up its superkey. For this reason, canOrderFrom is in BCNF.

stocks The stocks relation is in BCNF. The stocks relation has a primary key of sku. Since centerID is dependent on sku and the stocks relation

borrows both `sku` and `centerID` from other relations, `sku` is the superkey. For this reason, `stocks` is in BCNF.

`Chair` The `Chair` relation is in BCNF. The `Chair` relation is a subset of the `Item` relation. Each chair has one of each of its attributes that is strictly related to its `sku`. This makes the `sku` the superkey for the `Chair` relation. As such, since each functional dependency is dependent on `sku`, `Chair` is in BCNF.

`Table_` The `Table_` relation is in BCNF. The `Table_` relation is a subset of the `Item` relation. Each table has one of each of its attributes that is strictly related to its `sku`. This makes the `sku` the superkey for the `Table_` relation. As such, since each functional dependency is dependent on `sku`, `Table_` is in BCNF.

`Desk` The `Desk` relation is in BCNF. The `Desk` relation is a subset of the `Item` relation. Each desk has one of each of its attributes that is strictly related to its `sku`. This makes the `sku` the superkey for the `Desk` relation. As such, since each functional dependency is dependent on `sku`, `Desk` is in BCNF.

`Stool` The `Stool` relation is in BCNF. The `Stool` relation is a subset of the `Item` relation. Each stool has one of each of its attributes that is strictly related to its `sku`. This makes the `sku` the superkey for the `Stool` relation. As such, since each functional dependency is dependent on `sku`, `Stool` is in BCNF.

`Cabinet` The `Cabinet` relation is in BCNF. The `Cabinet` relation is a subset of the `Item` relation. Each cabinet has one of each of its attributes that is strictly related to its `sku`. This makes the `sku` the superkey for the `Cabinet` relation. As such, since each functional dependency is dependent on `sku`, `Cabinet` is in BCNF.

`Bedframe` The `Bedframe` relation is in BCNF. The `Bedframe` relation is a subset of the Item relation. Each bedframe has one of each of its attributes that is strictly related to its `sku`. This makes the `sku` the superkey for the `Bedframe` relation. As such, since each functional dependency is dependent on `sku`, `Bedframe` is in BCNF.

`features_Feature` This relation is in BCNF. The `features_Feature` relation has a key that it contains called `description`. This `description`, however long it may be, will be distinct meaning that each feature has a specific `description` fitting to that specific feature. This makes {`modelNumber`, `description`} the superkey for `features_Feature`. Also, since `count_` is dependent specifically on the description of that feature for that model, it is dependent on the superkey; this means that `features_Feature` is in BCNF.

5 Part 5

5.1 Relations

```sql
create domain posreal as double precision
check         (value > 0.0);

create domain posint as integer
check         (value > 0);

create table Supplier(supplierID  varchar(10),
                      name_       nchar varying(50)
                                  not null,
                      phone       varchar(12),
                      address     nchar varying(100),
                      country     char(2),
                      website     nchar varying(50),
                      primary key (supplierID));

create table Designer(designerID  varchar(10),
                      name_       nchar varying(50)
                                  not null,
                      phone       varchar(12),
                      address     nchar varying(100),
                      country     char(2),
                      website     nchar varying(50),
                      designFocus nchar varying(100),
                      primary key (designerID));

create table Set_(setID         varchar(10),
                  name_         nchar varying(50)
                                not null,
                  catalogYear   numeric(4,0),
                  catalogNumber integer
                                not null,
                  style_        nchar varying(30),
```

```
                  primary key   (setID));

create table Model(modelNumber varchar(10),
                   name_       nchar varying(50)
                               not null,
                   material    nchar varying(30),
                   upholstery  nchar varying(30),
                   durability  nchar varying(30),
                   color       nchar varying(30),
                   primary key (modelNumber));

create table Item(sku         varchar(10),
                  length_     posreal, -- in inches
                  width       posreal, -- in inches
                  height      posreal, -- in inches
                  condition   nchar varying(30),
                  weightLimit posreal, -- in pounds of weight
                  primary key (sku));

create table DistributionCenter(centerID    varchar(10),
                                name_       nchar varying(50)
                                            not null,
                                phone       varchar(12),
                                address     nchar varying(100),
                                country     char(2),
                                website     nchar varying(50),
                                primary key (centerID));

create table make(supplierID  varchar(10),
                  designerID  varchar(10),
                  setID       varchar(10),
                  primary key (supplierID,
                               designerID,
                               setID),
                  foreign key (supplierID)
                              references Supplier,
                  foreign key (designerID)
                              references Designer,
                  foreign key (setID)
                              references Set_);

create table contains_(setID       varchar(10),
                       modelNumber varchar(10),
                       count_      posint,
                       primary key (setID,
```

```
                                    modelNumber),
                      foreign key (setID)
                                  references Set_,
                      foreign key (modelNumber)
                                  references Model);

create table describes(modelNumber varchar(10)
                                  not null,
                      sku         varchar(10),
                      primary key (sku),
                      foreign key (modelNumber)
                                  references Model,
                      foreign key (sku)
                                  references Item);

create table canOrderFrom(centerID    varchar(10),
                          supplierID  varchar(10),
                          leadTime    double precision, -- in days
                          primary key (centerID,
                                       supplierID),
                          foreign key (centerID)
                                      references DistributionCenter,
                          foreign key (supplierID)
                                      references Supplier,
                          check       (leadTime >= 0.0));

create table stocks(centerID    varchar(10)
                                not null,
                    sku         varchar(10),
                    primary key (sku),
                    foreign key (centerID)
                                references DistributionCenter,
                    foreign key (sku)
                                references Item);

create table Chair(sku          varchar(10),
                   numberOfLegs posint,
                   hasCushion   boolean,
                   hasArms      boolean,
                   backHeight   posreal, -- in inches
                   seatHeight   posreal, -- in inches
                   primary key  (sku),
                   foreign key  (sku)
                                references Item);
```

```
create table Table_(sku           varchar(10),
                    numberOfLegs  posint,
                    numberOfSeats posint,
                    shape         nchar varying(30),
                    primary key   (sku),
                    foreign key   (sku)
                                  references Item);

create table Desk(sku             varchar(10),
                  angle           double precision,
                   ↪  -- in degrees, possibly negative
                  numberOfDrawers posint,
                  primary key     (sku),
                  foreign key     (sku)
                                  references Item,
                  check              (angle > -360.0
                                  and angle < 360.0));

create table Stool(sku          varchar(10),
                   numberOfLegs posint,
                   hasCushion   boolean,
                   hasSwivel    boolean,
                   primary key  (sku),
                   foreign key  (sku)
                                references Item);

create table Cabinet(sku                 varchar(10),
                     numberOfCompartments posint,
                     capacity             nchar varying(30),
                     primary key          (sku),
                     foreign key          (sku)
                                          references Item);

create table Bedframe(sku         varchar(10),
                      size_       nchar varying(30),
                      depth_      double precision,
                       ↪  -- in inches, possibly negative
                      primary key (sku),
                      foreign key (sku)
                                  references Item);

create table features_Feature(modelNumber varchar(10),
                              description nchar varying(50),
                              count_      posint,
                              primary key (modelNumber,
```

```
                                                 description),
                                foreign key (modelNumber)
                                             references Model);
```

5.2 Table Creation Statements

```
CREATE TABLE Supplier
  (
     supplierID VARCHAR(10) CHARACTER SET ASCII,
     name_      VARCHAR(50) CHARACTER SET utf8mb4 NOT NULL,
     phone      VARCHAR(12) CHARACTER SET ASCII,
     address    VARCHAR(100) CHARACTER SET utf8mb4,
     country    CHAR(2) CHARACTER SET ASCII DEFAULT 'US',
     website    VARCHAR(50) CHARACTER SET utf8mb4,
     PRIMARY KEY(supplierID),
     CHECK (country REGEXP '^[A-Z]{2}$'),
     CHECK (phone REGEXP '^[0-9]{7,12}$')
  );

CREATE TABLE Designer
  (
     designerID  VARCHAR(10) CHARACTER SET ASCII,
     name_       VARCHAR(50) CHARACTER SET utf8mb4 NOT NULL,
     phone       VARCHAR(12) CHARACTER SET ASCII,
     address     VARCHAR(100) CHARACTER SET utf8mb4,
     country     CHAR(2) CHARACTER SET ASCII DEFAULT 'US',
     website     VARCHAR(50) CHARACTER SET utf8mb4,
     designFocus VARCHAR(100) CHARACTER SET utf8mb4,
     PRIMARY KEY(designerID),
     CHECK (country REGEXP '^[A-Z]{2}$'),
     CHECK (phone REGEXP '^[0-9]{7,12}$')
  );

CREATE TABLE Set_
  (
     setID         VARCHAR(10) CHARACTER SET ASCII,
     name_         VARCHAR(50) CHARACTER SET utf8mb4 NOT NULL,
     catalogYear   DECIMAL(4, 0) UNSIGNED,
     catalogNumber BIGINT UNSIGNED ZEROFILL NOT NULL,
     style_        VARCHAR(30) CHARACTER SET utf8mb4,
     PRIMARY KEY(setID)
  );

CREATE TABLE Model
```

```sql
  (
     modelNumber VARCHAR(10) CHARACTER SET ASCII,
     name_       VARCHAR(50) CHARACTER SET utf8mb4 NOT NULL,
     material    VARCHAR(30) CHARACTER SET utf8mb4,
     upholstery  VARCHAR(30) CHARACTER SET utf8mb4,
     durability  VARCHAR(30) CHARACTER SET utf8mb4,
     color       VARCHAR(30) CHARACTER SET utf8mb4,
     PRIMARY KEY(modelNumber)
  );

CREATE TABLE Item
  (
     sku         VARCHAR(10) CHARACTER SET ASCII,
     length_     DOUBLE UNSIGNED,
     width       DOUBLE UNSIGNED,
     height      DOUBLE UNSIGNED,
     condition_  VARCHAR(30) CHARACTER SET utf8mb4,
     weightLimit DOUBLE UNSIGNED,
     PRIMARY KEY(sku),
     CHECK (length_ > 0.0),
     CHECK (width > 0.0),
     CHECK (height > 0.0),
     CHECK (weightLimit > 0.0)
  );

CREATE TABLE DistributionCenter
  (
     centerID VARCHAR(10) CHARACTER SET ASCII,
     name_    VARCHAR(50) CHARACTER SET utf8mb4 NOT NULL,
     phone    VARCHAR(12) CHARACTER SET ASCII,
     address  VARCHAR(100) CHARACTER SET utf8mb4,
     country  CHAR(2) CHARACTER SET ASCII DEFAULT 'US',
     website  VARCHAR(50) CHARACTER SET utf8mb4,
     PRIMARY KEY(centerID),
     CHECK (country REGEXP '^[A-Z]{2}$'),
     CHECK (phone REGEXP '^[0-9]{7,12}$')
  );

CREATE TABLE make
  (
     supplierID VARCHAR(10) CHARACTER SET ASCII,
     designerID VARCHAR(10) CHARACTER SET ASCII,
     setID      VARCHAR(10) CHARACTER SET ASCII,
     PRIMARY KEY(supplierID, designerID, setID),
     FOREIGN KEY(supplierID) REFERENCES Supplier(supplierID),
```

```sql
      FOREIGN KEY(designerID) REFERENCES Designer(designerID),
      FOREIGN KEY(setID) REFERENCES Set_(setID)
   );

CREATE TABLE contains_
   (
      setID       VARCHAR(10) CHARACTER SET ASCII,
      modelNumber VARCHAR(10) CHARACTER SET ASCII,
      count_      TINYINT UNSIGNED DEFAULT 1,
      PRIMARY KEY(setID, modelNumber),
      FOREIGN KEY(setID) REFERENCES Set_(setID),
      FOREIGN KEY(modelNumber) REFERENCES Model(modelNumber),
      CHECK (count_ > 0)
   );

CREATE TABLE describes
   (
      modelNumber VARCHAR(10) CHARACTER SET ASCII NOT NULL,
      sku         VARCHAR(10) CHARACTER SET ASCII,
      PRIMARY KEY(sku),
      FOREIGN KEY(modelNumber) REFERENCES Model(modelNumber),
      FOREIGN KEY(sku) REFERENCES Item(sku)
   );

CREATE TABLE canOrderFrom
   (
      centerID   VARCHAR(10) CHARACTER SET ASCII,
      supplierID VARCHAR(10) CHARACTER SET ASCII,
      leadTime   DOUBLE UNSIGNED,
      PRIMARY KEY(centerID, supplierID),
      FOREIGN KEY(centerID) REFERENCES DistributionCenter(centerID),
      FOREIGN KEY(supplierID) REFERENCES Supplier(supplierID)
   );

CREATE TABLE stocks
   (
      centerID VARCHAR(10) CHARACTER SET ASCII NOT NULL,
      sku      VARCHAR(10) CHARACTER SET ASCII,
      PRIMARY KEY(sku),
      FOREIGN KEY(centerID) REFERENCES DistributionCenter(centerID),
      FOREIGN KEY(sku) REFERENCES Item(sku)
   );

CREATE TABLE Chair
   (
```

```sql
    sku          VARCHAR(10) CHARACTER SET ASCII,
    numberOfLegs TINYINT UNSIGNED DEFAULT 4,
    hasCushion   BOOL DEFAULT false,
    hasArms      BOOL,
    backHeight   DOUBLE UNSIGNED,
    seatHeight   DOUBLE UNSIGNED,
    PRIMARY KEY(sku),
    FOREIGN KEY(sku) REFERENCES Item(sku),
    CHECK (numberOfLegs > 0),
    CHECK (backHeight > 0.0),
    CHECK (seatHeight > 0.0)
  );

CREATE TABLE Table_
  (
    sku           VARCHAR(10) CHARACTER SET ASCII,
    numberOfLegs  TINYINT UNSIGNED DEFAULT 4,
    numberOfSeats TINYINT UNSIGNED,
    shape         VARCHAR(30) CHARACTER SET utf8mb4,
    PRIMARY KEY(sku),
    FOREIGN KEY(sku) REFERENCES Item(sku),
    CHECK (numberOfLegs > 0),
    CHECK (numberOfSeats > 0)
  );

CREATE TABLE Desk
  (
    sku             VARCHAR(10) CHARACTER SET ASCII,
    angle           DOUBLE DEFAULT 0.0,
    numberOfDrawers TINYINT UNSIGNED,
    PRIMARY KEY(sku),
    FOREIGN KEY(sku) REFERENCES Item(sku),
    CHECK (angle > -360.0 AND angle < 360.0),
    CHECK (numberOfDrawers > 0)
  );

CREATE TABLE Stool
  (
    sku          VARCHAR(10) CHARACTER SET ASCII,
    numberOfLegs TINYINT UNSIGNED,
    hasCushion   BOOL,
    hasSwivel    BOOL,
    PRIMARY KEY(sku),
    FOREIGN KEY(sku) REFERENCES Item(sku),
    CHECK (numberOfLegs > 0)
```

```sql
  );

CREATE TABLE Cabinet
  (
     sku                  VARCHAR(10) CHARACTER SET ASCII,
     numberOfCompartments TINYINT UNSIGNED,
     capacity             VARCHAR(30) CHARACTER SET utf8mb4,
     PRIMARY KEY(sku),
     FOREIGN KEY(sku) REFERENCES Item(sku),
     CHECK (numberOfCompartments > 0)
  );

CREATE TABLE Bedframe
  (
     sku    VARCHAR(10) CHARACTER SET ASCII,
     size_  VARCHAR(30) CHARACTER SET utf8mb4,
     depth_ DOUBLE,
     PRIMARY KEY(sku),
     FOREIGN KEY(sku) REFERENCES Item(sku)
  );

CREATE TABLE features_Feature
  (
     modelNumber VARCHAR(10) CHARACTER SET ASCII,
     description VARCHAR(50) CHARACTER SET utf8mb4,
     count_      TINYINT UNSIGNED DEFAULT 1,
     PRIMARY KEY(modelNumber, description),
     FOREIGN KEY(modelNumber) REFERENCES Model(modelNumber),
     CHECK (count_ > 0)
  );
```

5.3 Data Counts

Supplier	50
Designer	50
Set_	100
Model	1000
Item	1000
DistributionCenter	50
make	1000
contains_	991
describes	1000
canOrderFrom	813
stocks	274

```
                    Chair  185
                   Table_  171
                     Desk  149
                    Stool  183
                  Cabinet  171
                 Bedframe  115
        features_Feature  8351
```

5.4 Sample Interactions

```
mysql> explain Bedframe;
+--------+-------------+------+-----+---------+-------+
| Field  | Type        | Null | Key | Default | Extra |
+--------+-------------+------+-----+---------+-------+
| sku    | varchar(10) | NO   | PRI | NULL    |       |
| size_  | varchar(30) | YES  |     | NULL    |       |
| depth_ | double      | YES  |     | NULL    |       |
+--------+-------------+------+-----+---------+-------+
3 rows in set (0.00 sec)

mysql> insert into Bedframe values (('6yNuvTHGL9', 'double twin', 5.9));
ERROR 1136 (21S01): Column count doesn't match value count at row 1
mysql> insert into Bedframe values ('6yNuvTHGL9', 'double twin', 5.9);
ERROR 1452 (23000): Cannot add or update a child row: a foreign key constraint
 ↪  fails (`aaltman`.`bedframe`, CONSTRAINT `bedframe_ibfk_1` FOREIGN KEY (`sku`)
 ↪  REFERENCES `Item` (`sku`))
mysql> explain Item;
+-------------+-----------------+------+-----+---------+-------+
| Field       | Type            | Null | Key | Default | Extra |
+-------------+-----------------+------+-----+---------+-------+
| sku         | varchar(10)     | NO   | PRI | NULL    |       |
| length_     | double unsigned | YES  |     | NULL    |       |
| width       | double unsigned | YES  |     | NULL    |       |
| height      | double unsigned | YES  |     | NULL    |       |
| condition_  | varchar(30)     | YES  |     | NULL    |       |
| weightLimit | double unsigned | YES  |     | NULL    |       |
+-------------+-----------------+------+-----+---------+-------+
6 rows in set (0.00 sec)

mysql> insert into Item values ('6yNuvTHGL9', 120.0, 70.0, 40.5, 'like new',
 ↪  276.89);
Query OK, 1 row affected (0.01 sec)

mysql> insert into Bedframe values ('6yNuvTHGL9', 'double twin', 5.9);
```

```
Query OK, 1 row affected (0.00 sec)

mysql> select * from Item;
+------------+---------+-------+--------+------------+-------------+
| sku        | length_ | width | height | condition_ | weightLimit |
+------------+---------+-------+--------+------------+-------------+
| 6yNuvTHGL9 |     120 |    70 |   40.5 | like new   |      276.89 |
+------------+---------+-------+--------+------------+-------------+
1 row in set (0.00 sec)

mysql> select * from Bedframe;
+------------+-------------+--------+
| sku        | size_       | depth_ |
+------------+-------------+--------+
| 6yNuvTHGL9 | double twin |    5.9 |
+------------+-------------+--------+
1 row in set (0.00 sec)

mysql> delete from Bedframe select * from Bedframe;
ERROR 1064 (42000): You have an error in your SQL syntax; check the manual that
 ↪  corresponds to your MySQL server version for the right syntax to use near
 ↪  'select * from Bedframe' at line 1
mysql> delete from Bedframe;
Query OK, 1 row affected (0.00 sec)

mysql> delete from Item;
Query OK, 1 row affected (0.01 sec)
```

- We learned the proper syntax for `INSERT` and `DELETE` statements.
- We learned not to insert a value in a table that has a foreign key constraint before that constraint is satisfied.

5.5 Data Sources

We generated our data randomly, using the following C++ program (`random_gen/random_gen.cpp`), along with some initial data files drawn from `https://www.random.org`:

```cpp
#include <cstdio>
#include <cstdlib>
#include <ctime>
#include <fstream>
#include <iostream>
#include <string>
```

```
using namespace std;

int main() {
  srand((unsigned int)time(NULL));

  int i = 0;

  string sku[2000];

  ifstream skuList_file("skuList.txt");

  ifstream setIDs_file("setIDs.txt");

  ofstream set_data;
  ofstream output;
  ofstream contains_file;
  ofstream chairs_output;
  ofstream tables_output;
  ofstream desks_output;
  ofstream stools_output;
  ofstream cabinets_output;
  ofstream bedframes_output;
  ofstream make_file;
  ofstream canOrderFrom_file;
  string line;
  ifstream modelNumbers_file("modelNumbers.txt");
  string name1[18];
  string name2[18];
  string name3[6];
  string material[26];
  string upholstery[13];
  string color[25];
  string durability[10];
  string modelNumbers[1000];

  ifstream styles_file("styles.txt");

  string setIDs[100];
  string styles[100];

  if (styles_file.is_open()) {
    while (getline(styles_file, line)) {
      styles[i] = line;
      i++;
      // cout << line << endl;
```

```
    }
    styles_file.close();
  } else
    cerr << "Unable to open styles file" << endl;

  name1[0] = "Timeless";
  name1[1] = "Futuristic";
  name1[2] = "Contemporary";
  name1[3] = "Homely";
  name1[4] = "Ancient";
  name1[5] = "Stylish";
  name1[6] = "Eccentric";
  name1[7] = "Large";
  name1[8] = "Aged";
  name1[9] = "Strong";
  name1[10] = "Eclectic";
  name1[11] = "Rustic";
  name1[12] = "Modern";
  name1[13] = "Revolutionary";
  name1[14] = "Indian";
  name1[15] = "German";
  name1[16] = "Italian";
  name1[17] = "Norwegian";

  name2[0] = "Angular";
  name2[1] = "Rounded";
  name2[2] = "Jagged";
  name2[3] = "Hefty";
  name2[4] = "Light";
  name2[5] = "Chunky";
  name2[6] = "Macho";
  name2[7] = "Thin";
  name2[8] = "Badass";
  name2[9] = "Ladylike";
  name2[10] = "Slender";
  name2[11] = "Wimpy";
  name2[12] = "Oblique";
  name2[13] = "Bowed";
  name2[14] = "Gaunt";
  name2[15] = "Meager";
  name2[16] = "Fat";
  name2[17] = "Genteel";

  name3[0] = "Chair";
  name3[1] = "Table";
```

```
    name3[2] = "Desk";
    name3[3] = "Stool";
    name3[4] = "Cabinet";
    name3[5] = "Bedframe";

    material[0] = "Ash";
    material[1] = "Cherry";
    material[2] = "Maple";
    material[3] = "Birch";
    material[4] = "Teak";
    material[5] = "Hickory";
    material[6] = "Oak";
    material[7] = "Walnut";
    material[8] = "Aluminum";
    material[9] = "Steel";
    material[10] = "Beech";
    material[11] = "Alder";
    material[12] = "Elm";
    material[13] = "Pine";
    material[14] = "Cottonwood";
    material[15] = "Hemlock";
    material[16] = "Fir";
    material[17] = "Cedar";
    material[18] = "Balsa";
    material[19] = "Magnesium Alloy";
    material[20] = "Coast Redwood";
    material[21] = "Afzelia";
    material[22] = "Ebony";
    material[23] = "Lindens";
    material[24] = "Purpleheart";
    material[25] = "Aspen";

    upholstery[0] = "Linen";
    upholstery[1] = "Leather";
    upholstery[2] = "Cotton";
    upholstery[3] = "Wool";
    upholstery[4] = "Cotton Blend";
    upholstery[5] = "Vinyl";
    upholstery[6] = "Silk";
    upholstery[7] = "Acetate";
    upholstery[8] = "Acrylic";
    upholstery[9] = "Nylon";
    upholstery[10] = "Olefin";
    upholstery[11] = "Polyester";
    upholstery[12] = "Rayon";
```

```
142
143     color[0] = "Blue";
144     color[1] = "Green";
145     color[2] = "Red";
146     color[3] = "White";
147     color[4] = "Black";
148     color[5] = "Yellow";
149     color[6] = "Grey";
150     color[7] = "Orange";
151     color[8] = "Purple";
152     color[9] = "Pink";
153     color[10] = "Violet";
154     color[11] = "Magenta";
155     color[12] = "Gold";
156     color[13] = "Cyan";
157     color[14] = "Turquoise";
158     color[15] = "Lavender";
159     color[16] = "Maroon";
160     color[17] = "Olive";
161     color[18] = "Indigo";
162     color[19] = "Tan";
163     color[20] = "Salmon";
164     color[21] = "Sky Blue";
165     color[22] = "Teal";
166     color[23] = "Coral";
167     color[24] = "Silver";
168
169     durability[0] = "Very Strong";
170     durability[1] = "Strong";
171     durability[2] = "Somewhat Strong";
172     durability[3] = "Very Sturdy";
173     durability[4] = "Sturdy";
174     durability[5] = "Somewhat Sturdy";
175     durability[6] = "Wobbly";
176     durability[7] = "Somewhat Wobbly";
177     durability[8] = "Very Wobbly";
178     durability[9] = "Indestructable";
179
180     output.open("model_data.csv");
181
182     chairs_output.open("chairs.csv");
183     tables_output.open("tables.csv");
184     desks_output.open("desks.csv");
185     stools_output.open("stools.csv");
186     cabinets_output.open("cabinets.csv");
```

```
  bedframes_output.open("bedframes.csv");

  string item;
  string preName;
  string postName;
  string current_material;
  string current_upholstery;
  string current_durability;
  string current_color;

  i = 0;

  if (skuList_file.is_open()) {
    while (getline(skuList_file, line)) {
      sku[i] = line;
      i++;
    }
    skuList_file.close();
  } else
    cerr << "Unable to open skuList file" << endl;

  i = 0;
  // Write the modelNumbers into a file
  if (modelNumbers_file.is_open()) {
    while (getline(modelNumbers_file, line)) {
      modelNumbers[i] = line;

      int current_item = rand() % 6;

      preName = name1[rand() % 18];
      postName = name2[rand() % 18];
      current_material = material[rand() % 25];
      current_upholstery = upholstery[rand() % 12];
      current_durability = durability[rand() % 10];
      current_color = color[rand() % 25];
      item = name3[current_item];

      switch (current_item + 1) {
      case 1:
        chairs_output << line << ",";
        chairs_output << sku[i] << ",";
        i++;
        chairs_output << rand() % 3 + 3 << ",";
        chairs_output << rand() % 2 << ",";
        chairs_output << rand() % 2 << ",";
```

```
        chairs_output << rand() % 12 + 24 << ",";
        chairs_output << rand() % 24 + 24 << endl;
        break;
      case 2:
        tables_output << line << ",";
        tables_output << sku[i] << ",";
        i++;
        tables_output << rand() % 6 + 4 << ",";
        tables_output << rand() % 8 + 4 << ",";
        if (rand() % 2 == 1) {
          tables_output << "Round" << endl;
        } else {
          tables_output << "Rectangular" << endl;
        }
        break;
      case 3:
        desks_output << line << ",";
        desks_output << sku[i];
        i++;
        desks_output << rand() % 45 + 0 << ",";
        desks_output << rand() % 6 + 1 << endl;
        break;
      case 4:
        stools_output << line << ",";
        stools_output << sku[i] << ",";
        i++;
        stools_output << rand() % 3 + 3 << ",";
        stools_output << rand() % 2 << ",";
        stools_output << rand() % 2 << endl;
        break;
      case 5:
        cabinets_output << line << ",";
        cabinets_output << sku[i] << ",";
        i++;
        cabinets_output << rand() % 6 + 1 << ",";
        cabinets_output << rand() % 10 + 1 << endl;
        break;
      case 6:
        bedframes_output << line << ",";
        bedframes_output << sku[i] << ",";
        switch (rand() % 6 + 1) {
        case 1:
          bedframes_output << "Twin"
                           << ",";
          break;
```

```
        case 2:
          bedframes_output << "Twin XL"
                           << ",";
          break;
        case 3:
          bedframes_output << "Full"
                           << ",";
          break;
        case 4:
          bedframes_output << "Queen"
                           << ",";
          break;
        case 5:
          bedframes_output << "King"
                           << ",";
          break;
        case 6:
          bedframes_output << "California King"
                           << ",";
          break;
        default:
          cerr << "bad case in switch statement!!!" << endl;
        }
        bedframes_output << rand() % 24 + 24 << endl;
        break;
      default:
        cerr << "bad case in switch statement!!!" << endl;
      }

      output << line << ",";
      output << preName << " " << postName << " " << item << ",";
      output << current_material << ",";
      output << current_upholstery << ",";
      output << current_durability << ",";
      output << current_color << endl;
    }
    modelNumbers_file.close();
  } else
    cerr << "Unable to open modelNumbers file";

  ofstream output_file;

  ifstream designerIDs_file("designerIDs.txt");
  ifstream firstNames_file("firstNames.txt");
  ifstream lastNames_file("lastNames.txt");
```

```
ifstream phones_file("phones.txt");
ifstream designerAddresses_file("designerAddresses.txt");
ifstream countries_file("countries.txt");
ifstream designFocuses_file("designFocuses.txt");

string designerIDs[50];
string firstNames[50];
string lastNames[50];
string phones[50];
string designerAddresses[50];
string countries[249];
string websites[50];
string designFocuses[11];
string domainSuffix[5];

domainSuffix[0] = ".com";
domainSuffix[1] = ".net";
domainSuffix[2] = ".org";
domainSuffix[3] = ".biz";
domainSuffix[4] = ".info";

i = 0;

// Read "designerIDs.txt" into string array

if (designerIDs_file.is_open()) {
  while (getline(designerIDs_file, line)) {
    designerIDs[i] = line;
    i++;
  }
  designerIDs_file.close();
} else
  cerr << "Unable to open ID file" << endl;

i = 0;

// Read "firstNames.txt" into string array

if (firstNames_file.is_open()) {
  while (getline(firstNames_file, line)) {
    firstNames[i] = line;
    i++;
  }
  firstNames_file.close();
} else
```

```
    cerr << "Unable to open firstNames file" << endl;

  i = 0;

  // Read "lastNames.txt" into string array

  if (lastNames_file.is_open()) {
    while (getline(lastNames_file, line)) {
      lastNames[i] = line;
      i++;
    }
    lastNames_file.close();
  } else
    cerr << "Unable to open lastNames file" << endl;

  i = 0;

  // Read "phones.txt" into string array

  if (phones_file.is_open()) {
    while (getline(phones_file, line)) {
      phones[i] = line;
      i++;
    }
    phones_file.close();
  } else
    cerr << "Unable to open phones file" << endl;

  i = 0;

  // Read "designerAddresses.txt into string array

  if (designerAddresses_file.is_open()) {
    while (getline(designerAddresses_file, line)) {
      designerAddresses[i] = line;
      i++;
    }
    designerAddresses_file.close();
  } else
    cerr << "Unable to open Addresses file" << endl;

  i = 0;

  // Read "countries.txt" into string array
  if (countries_file.is_open()) {
```

```
    while (getline(countries_file, line)) {
      countries[i] = line;
      i++;
    }
    countries_file.close();
  } else
    cerr << "Unable to open countries file" << endl;

  i = 0;

  // Create website addresses based on names, save them into websites array
  while (i < 50) {
    websites[i] =
        "www." + firstNames[i] + lastNames[i] + domainSuffix[rand() % 5];
    i++;
  }

  i = 0;

  // Read "designFocuses.txt" into string array

  if (designFocuses_file.is_open()) {
    while (getline(designFocuses_file, line)) {
      designFocuses[i] = line;
      i++;
    }
    designFocuses_file.close();
  } else
    cerr << "Unable to open designFocuses file" << endl;

  i = 0;
  // Write to output file
  // use a loop to write output to a file
  // SUBTASKS
  // write designerID on a line, end line
  // write a first name then a last name on a line, end line
  // write a phone number on a line, end line
  // write an address on a line, end line
  // write a random country on a line, end line
  // write a website on a line, end line
  // write a random design focus on a line, end line
  // end line for spacing
  output_file.open("designer_data.csv");

  while (i < 50) {
```

```
    output_file << designerIDs[i] << ",";
    output_file << firstNames[i] << " " << lastNames[i] << ",";
    output_file << phones[i] << ",";
    output_file << "\"" << designerAddresses[i] << "\""
                << ",";
    output_file << countries[rand() % 249] << ",";
    output_file << websites[i] << ",";
    output_file << designFocuses[rand() % 11] << endl;
    i++;
  }
  output_file.close();

  i = 0;

  string supplierIDs[50];
  string supplierNames[50];
  string supplierWebsiteNames[50];
  string addresses[50];

  // supplierIDs.txt
  ifstream supplierIDs_file("supplierIDs.txt");
  // names.txt
  ifstream supplierNames_file("supplierNames.txt");
  // supplierWebsiteNames.txt
  ifstream supplierWebsiteNames_file("supplierWebsiteNames.txt");
  // addresses.txt
  ifstream addresses_file("addresses.txt");

  // Read "supplierIDs.txt" into an array

  if (supplierIDs_file.is_open()) {
    while (getline(supplierIDs_file, line)) {
      supplierIDs[i] = line;
      i++;
    }
    supplierIDs_file.close();
  } else
    cerr << "Unable to open supplierIDs file" << endl;

  i = 0;

  // Read "supplierNames.txt" to names array

  if (supplierNames_file.is_open()) {
    while (getline(supplierNames_file, line)) {
```

```
      supplierNames[i] = line;
      i++;
    }
    supplierNames_file.close();
  } else
    cerr << "Unable to open names file" << endl;

  i = 0;

  // read "supplierWebsiteNames.txt" into an array

  if (supplierWebsiteNames_file.is_open()) {
    while (getline(supplierWebsiteNames_file, line)) {
      supplierWebsiteNames[i] = line;
      i++;
    }
    supplierWebsiteNames_file.close();
  } else
    cerr << "Unable to open supplierWebsiteNames file" << endl;

  i = 0;

  // Read "addresses.txt" to addresses array

  if (addresses_file.is_open()) {
    while (getline(addresses_file, line)) {
      addresses[i] = line;
      i++;
    }
    addresses_file.close();
  } else
    cerr << "Unable to open addresses file" << endl;

  i = 0;

  // output data to supplier_data.txt
  // supplierIDs
  // name
  // phone
  // address
  // country
  // website
  output_file.open("supplier_data.csv");
  while (i < 50) {
    output_file << supplierIDs[i] << ",";
```

```
    output_file << supplierNames[i] << ",";
    output_file << phones[i] << ",";
    output_file << "\"" << addresses[i] << "\""
                << ",";
    output_file << countries[rand() % 249] << ",";
    output_file << "www." << supplierWebsiteNames[i]
                << domainSuffix[rand() % 5];
    output_file << endl;
    i++;
  }
  output_file.close();

  i = 0;

  string centerIDs[50];
  string centerNames[50];
  string centerWebsiteNames[50];

  // centerIDs.txt
  ifstream centerIDs_file("centerIDs.txt");
  // names.txt
  ifstream centerNames_file("centerNames.txt");
  // centerWebsiteNames.txt
  ifstream centerWebsiteNames_file("centerWebsiteNames.txt");

  // Read "centerIDs.txt" into an array

  if (centerIDs_file.is_open()) {
    while (getline(centerIDs_file, line)) {
      centerIDs[i] = line;
      i++;
    }
    centerIDs_file.close();
  } else
    cerr << "Unable to open centerIDs file" << endl;

  i = 0;

  // Read "centerNames.txt" to names array

  if (centerNames_file.is_open()) {
    while (getline(centerNames_file, line)) {
      centerNames[i] = line;
      i++;
    }
```

```
    centerNames_file.close();
  } else
    cerr << "Unable to open names file" << endl;

  i = 0;

  // read "centerWebsiteNames.txt" into an array

  if (centerWebsiteNames_file.is_open()) {
    while (getline(centerWebsiteNames_file, line)) {
      centerWebsiteNames[i] = line;
      i++;
    }
    centerWebsiteNames_file.close();
  } else
    cerr << "Unable to open centerWebsiteNames file" << endl;

  i = 0;

  // output data to center_data.txt
  // centerIDs
  // name
  // phone
  // address
  // country
  // website
  output_file.open("center_data.csv");
  while (i < 50) {
    output_file << centerIDs[i] << ",";
    output_file << centerNames[i] << ",";
    output_file << phones[i] << ",";
    output_file << "\"" << addresses[i] << "\""
                << ",";
    output_file << countries[rand() % 249] << ",";
    output_file << "www." << centerWebsiteNames[i] << domainSuffix[rand() % 5];
    output_file << endl;
    i++;
  }
  output_file.close();

  ifstream modelNumber_file("modelNumbers.txt");

  ofstream modelNumberToSku_file;
  ofstream item_file;

```

```
i = 0;

// Take in "modelNumbers.txt" and put into array

if (modelNumber_file.is_open()) {
  while (getline(modelNumber_file, line)) {
    modelNumbers[i] = line;
    i++;
  }
  modelNumber_file.close();
} else
  cerr << "Unable to open modelNumber file" << endl;

i = 0;
// output a file that assigns each modelNumber an sku

modelNumberToSku_file.open("modelNumberToSku.csv");

while (i < 1000) {
  modelNumberToSku_file << modelNumbers[i] << "," << sku[i] << endl;
  i++;
}
i = 0;

// output "item_data.csv"

item_file.open("item_data.csv");

while (i < 1000) {
  item_file << sku[i] << "," << rand() % 24 + 48 << "," << rand() % 24 + 48
            << "," << rand() % 24 + 48 << ","
            << "New"
            << "," << rand() % 200 + 300 << endl;
  i++;
}
i = 0;

if (setIDs_file.is_open()) {
  while (getline(setIDs_file, line)) {
    setIDs[i] = line;
    i++;
  }
  setIDs_file.close();
} else
  cerr << "Unable to open setIDs file" << endl;
```

```
682
683     i = 0;
684
685     set_data.open("set_data.csv");
686
687     while (i < 100) {
688       set_data << setIDs[i] << ",";
689       set_data << styles[i] << " set"
690                << ",";
691       set_data << rand() % 33 + 1985 << ",";
692       set_data << i << ",";
693       set_data << styles[i] << endl;
694       i++;
695     }
696
697     contains_file.open("contains_data.csv");
698     i = 0;
699     while (i < 1000) {
700       contains_file << setIDs[rand() % 100] << ",";
701       contains_file << modelNumbers[rand() % 1000] << ",";
702       contains_file << (rand() % 6 + 1) << endl;
703       i++;
704     }
705     contains_file.close();
706
707     i = 0;
708
709     make_file.open("make_data.csv");
710
711     while (i < 1000) {
712       make_file << supplierIDs[rand() % 50] << ",";
713       make_file << designerIDs[rand() % 50] << ",";
714       make_file << setIDs[rand() % 100] << endl;
715       i++;
716     }
717
718     make_file.close();
719
720     i = 0;
721
722     canOrderFrom_file.open("canOrderFrom_data.csv");
723
724     while (i < 1000) {
725       canOrderFrom_file << centerIDs[rand() % 50] << ",";
726       canOrderFrom_file << supplierIDs[rand() % 50] << ",";
```

```
727       canOrderFrom_file << (rand() % 30 + 1) << endl;
728
729       i++;
730     }
731     canOrderFrom_file.close();
732
733     ofstream stocks_file;
734
735     stocks_file.open("stocks_data.csv");
736
737     for (int l = 0; l < 50; l++) {
738       int count = rand() % 15 + 1;
739       for (int x = 1; x < count; x++) {
740         stocks_file << centerIDs[l] << "," << sku[rand() % 1000] << endl;
741       }
742     }
743     stocks_file.close();
744
745     string features[10];
746
747     features[0] = "Fancy Knobs";
748     features[1] = "Carved Inlays";
749     features[2] = "Ivory Handles";
750     features[3] = "Claw Feet";
751     features[4] = "Gold Inlaid Designs";
752     features[5] = "Fancy Molding";
753     features[6] = "Gold Hinges";
754     features[7] = "Padded Feet";
755     features[8] = "Studded Corners";
756     features[9] = "Textured";
757     ofstream features_file;
758
759     features_file.open("features_data.csv");
760
761     for (int l = 0; l < 1000; l++) {
762       for (int x = 0; x < 10; x++) {
763         int count = rand() % 6;
764         if (count > 0) {
765           features_file << modelNumbers[l] << "," << features[x] << "," << count
766                         << endl;
767         }
768       }
769     }
770
771     features_file.close();
```

```

  return 0;
}
```

We used the command syntax `load data local infile {CSV file name} ignore into table {table name} fields terminated by ',' optionally enclosed by '"'`; to load the data into each table from the corresponding CSV file, which took care of duplicate entries by ignoring them; all other constraints were satisfied by construction due to the way the generation program was written and run.

5.6 Data Samples

```
mysql> select * from Supplier limit 10;
+------------+-----------------------------------+------------+-------------------+---------------------------+---------+-----------------------------------+
| supplierID | name_                             | phone      | address           | country | website                 |
+------------+-----------------------------------+------------+-------------------+---------------------------+---------+-----------------------------------+
| 06DeQdaf4X | The Snowy Pencil Leather Company   | 8039090134 | 379 Virginia St. Niagara Falls, NY 14304      | TR      | www.TheSnowyPencilLeatherCompany.biz |
| 0OaJOGFGdD | The Cloudy Chicken Fabrics Company | 8749357327 | 8247 S. Hamilton Drive Cedar Rapids, IA 52402 | TO      | www.TheCloudyChickenFabricsCompany.info |
| 0x1V8UVLqh | The Deep Lamp Lumber Company       | 7463153612 | 72 Somerset Lane Ypsilanti, MI 48197          | AQ      | www.TheDeepLampLumberCompany.info |
| 3H7BBv3j7C | Tall Mouse Supplier                | 4123072387 | 9444 Blackburn Lane Wenatchee, WA 98801       | PA      | www.TallMouseSupplier.org |
| 3LnmekGoPa | The Happy Chinchilla Metal Company | 9115174479 | 50 Catherine Lane Huntsville, AL 35803        | ZM      | www.TheHappyChinchillaMetalCompany.com |
| 45AVHG6SDL | Big Light-Switch Metal             | 2219847475 | 9916 Bridgeton Ave. Austin, MN 55912          | AQ      | www.BigLight-SwitchMetal.org |
| 4hGbj2aBVR | Red Baboon Metal                   | 8433984003 | 860 Glen Ridge Rd. Whitestone, NY 11357       | BT      | www.RedBaboonMetal.net |
```

```
| 4xp0ETTkc8 | The White Skunk Metal Company      | 2695261566 | 52 Deerfield Lane Woodhaven, NY 11421         | DJ      | www.TheWhiteSkunkMetalCompany.info |
| 5Hq7fF9aK0 | The Opaque Pear Supplier Company   | 9201886129 | 7254 Hickory Ave. Centreville, VA 20120       | CI      | www.TheOpaquePearSupplierCompany.org |
| 5SKHZGJsjV | Transparent Cherry Lumber          | 4623416776 | 1 Redwood Rd. Roanoke, VA 24012               | ZW      | www.TransparentCherryLumber.org |
+------------+------------------------------------+------------+-------------------------------+---------+-------------------------------------+
10 rows in set (0.00 sec)

mysql> select * from Designer limit 10;
+------------+------------------+------------+-------------------------------------+--------------+---------+-------------------------+--------------+
| designerID | name_            | phone      | address                             | country | website                 | designFocus  |
| 3lejckzNYS | Lyndia Butler    | 7034816719 | 241 Morris Dr. Bowling Green, KY 42101            | PE      | www.LyndiaButler.org    | Mid-Century  |
| 4MVbu2iI15 | Mellissa Rich    | 4623416776 | 4 Garden Rd. Dunedin, FL 34698                    | AW      | www.MellissaRich.biz    | Scandinavian |
| 7RYpyw9es0 | Casimira Carney  | 1437523438 | 8498 Young Street Oklahoma City, OK 73112         | ML      | www.CasimiraCarney.com  | Industrial   |
| 7ZZbCsXnv0 | Charmaine Pineda | 4204843909 | 497 Young Lane Panama City, FL 32404              | NL      | www.CharmainePineda.org | Rococo       |
| A7oA1v9Ax1 | Renee Holmes     | 2589423308 | 78 Santa Clara Drive Huntsville, AL 35803         | ZW      | www.ReneeHolmes.org     | Modern       |
| AZadqlHsUN | Elia Melton      | 7421854946 | 519 Bayberry Ave. Bayside, NY 11361               | SX      | www.EliaMelton.org      | Industrial   |
| b9yOGUx3pl | Jeni Wilson      | 5416771400 | 215 Birchwood Ave. Boston, MA 02127               | KR      | www.JeniWilson.net      | Industrial   |
| bBMMbiXWX2 | Ludivina Hunt    | 7071474222 | 9120 Santa Clara St. Huntington Station, NY 11746 | GP      | www.LudivinaHunt.net    | Rococo       |
| bDcoTRYgku | Sierra Novak     | 2695261566 | 155 N. Elm Street Rego Park, NY 11374             | VN      | www.SierraNovak.org     | Scandinavian |
| BejwYSNzm7 | Claribel Vasquez | 6712497760 | 256 Water Ave. Shelton, CT 06484                  | LA      | www.ClaribelVasquez.org | Eclecticism  |
+------------+------------------+------------+-------------------------------------+--------------+---------+-------------------------+--------------+
10 rows in set (0.00 sec)
```

```
mysql> select * from Set_ limit 10;
+------------+-----------------+-------------+----------------------+-------------+
| setID      | name_           | catalogYear | catalogNumber        | style_      |
+------------+-----------------+-------------+----------------------+-------------+
| 0naFKFb6se | dizzy set       |        2013 | 00000000000000000029 | dizzy       |
| 0nZQJHrEtZ | fowl set        |        1992 | 00000000000000000013 | fowl        |
| 0To5uOwhgK | spectacular set |        1986 | 00000000000000000040 | spectacular |
| 1JtLYWK555 | receive set     |        1985 | 00000000000000000070 | receive     |
| 5JnQxcSlvJ | earthquake set  |        1985 | 00000000000000000066 | earthquake  |
| 6eHdxhCYK7 | fish set        |        2001 | 00000000000000000000 | fish        |
| 73l1iRUEvw | capable set     |        2014 | 00000000000000000071 | capable     |
| 73X8UvpzrJ | slip set        |        1998 | 00000000000000000093 | slip        |
| 8blchiMxYL | right set       |        2000 | 00000000000000000038 | right       |
| 980pBKGuMU | file set        |        2003 | 00000000000000000083 | file        |
+------------+-----------------+-------------+----------------------+-------------+
10 rows in set (0.00 sec)

mysql> select * from Model limit 10;
+-------------+--------------------------+----------+------------+----------------
-+---------+
| modelNumber | name_                    | material | upholstery | durability
   | color   |
+-------------+--------------------------+----------+------------+----------------
-+---------+
| 00aWzohu8g  | Ancient Genteel Desk     | Ebony    | Wool       | Very Strong
   | Orange  |
| 02SVZ28q47  | Ancient Angular Stool    | Hemlock  | Nylon      | Very Wobbly
   | Silver  |
| 0aBIk36Hs2  | Contemporary Hefty Chair | Afzelia  | Acetate    | Very Strong
   | Indigo  |
| 0JCfsD5Pck  | Futuristic Rounded Stool | Maple    | Linen      | Sturdy
   | Grey    |
| 0KSfx7M1IN  | Contemporary Fat Desk    | Elm      | Cotton     | Very Wobbly
   | Teal    |
| 0KSjIO5LDu  | Eclectic Bowed Desk      | Cherry   | Cotton     | Sturdy
   | Black   |
| 0mp9PGPdKw  | Futuristic Bowed Bedframe | Balsa    | Linen      | Indestructable
   | Gold    |
| 0T2PaDHCEw  | Contemporary Light Chair | Pine     | Acetate    | Somewhat
  Wobbly | Gold    |
| 0uzi2lU3dA  | Large Wimpy Table        | Fir      | Polyester  | Very Sturdy
   | Olive   |
| 0vhyAaLCYg  | Italian Rounded Bedframe | Pine     | Cotton     | Indestructable
   | Magenta |
```

```
+-------------+--------------------------+----------+------------+----------------+---------+
10 rows in set (0.00 sec)

mysql> select * from Item limit 10;
+------------+---------+-------+--------+------------+-------------+
| sku        | length_ | width | height | condition_ | weightLimit |
+------------+---------+-------+--------+------------+-------------+
| 09S4mRvuGE |      61 |    66 |     51 | New        |         309 |
| 0EbNx7hvlO |      64 |    58 |     48 | New        |         415 |
| 0EKsH99oc8 |      64 |    67 |     58 | New        |         342 |
| 0eLPBgmxXc |      67 |    57 |     53 | New        |         340 |
| 0F6T8XT2R1 |      50 |    59 |     53 | New        |         310 |
| 0hMisTAJ6K |      66 |    60 |     68 | New        |         311 |
| 0RzRFaHBm4 |      48 |    50 |     69 | New        |         350 |
| 0snow42N9H |      62 |    52 |     56 | New        |         325 |
| 0t5AqpGvtk |      63 |    65 |     65 | New        |         372 |
| 0WxtLk4cbX |      57 |    49 |     54 | New        |         302 |
+------------+---------+-------+--------+------------+-------------+
10 rows in set (0.00 sec)

mysql> select * from DistributionCenter limit 10;
+------------+---------------------------------------------+------------+---------+
| centerID   | name_                                       | phone      | address
                                     | country | website
                     |
+------------+---------------------------------------------+------------+---------+
| 0gkU23fhbT | Sub-Zero Duck Warehouse                     | 2738728093 | 7083
  Green Court Pembroke Pines, FL 33028   | LA      |
  www.Sub-ZeroDuckWarehouse.com                    |
| 0mslNHYadG | The Complicated Hamster Distribution Company | 8256624098 | 904
  Creekside St. Pottstown, PA 19464       | UG      |
  www.TheComplicatedHamsterDistributionCompany.net |
| 3D5zIxPUZv | Opaque Frog Storage                         | 1046485315 | 92 W.
  Princeton Rd. Long Beach, NY 11561    | KM      | www.OpaqueFrogStorage.info
                      |
| 4XdNKanfY1 | The White Pear Warehouse Company            | 8485622783 | 9810
  Edgefield St. Natick, MA 01760         | WS      |
  www.TheWhitePearWarehouseCompany.com             |
```

```
| 5RDZWNp0Ym | The Acute Turtle Warehouse Company           | 7003923806 | 99 Lower River Drive Mount Vernon, NY 10550 | KE      |
| 8ECOfFZSgC | The Grey Squirrel Distribution Company       | 1437523438 | 827 Dogwood Ave. Saginaw, MI 48601          | LC      |
| a1Xzp6g1lw | Cold Fish Distributing                       | 8511426114 | 69 Bay Meadows Lane Bridgeton, NJ 08302     | KG      | www.ColdFishDistributing.com |
| aD4TjXPcH4 | Joey Distribution                            | 5911329769 | 937 Cypress Street Butte, MT 59701          | NU      | www.JoeyDistribution.org |
| At7FkrAz5m | Piping Bull Distributing                     | 9201886129 | 7254 Hickory Ave. Centreville, VA 20120     | EG      |
| beYPqnGi9f | Sub-Zero Lemon Warehouse                     | 7463153612 | 72 Somerset Lane Ypsilanti, MI 48197        | FJ      |
+------------+----------------------------------------------+------------+-----------------------------------------+---------+-------------------------------+
10 rows in set (0.00 sec)

mysql> select * from make limit 10;
+------------+------------+------------+
| supplierID | designerID | setID      |
+------------+------------+------------+
| 3H7BBv3j7C | 3lejckzNYS | jYIzpoPJ0v |
| 4hGbj2aBVR | 3lejckzNYS | pzZzDyittU |
| 5SKHZGJsjV | 3lejckzNYS | ajPkfYW6Rd |
| Dlq94L7rPh | 3lejckzNYS | LPi87bcU5k |
| dWuUGFOe1b | 3lejckzNYS | Auae4YJ0eg |
| ldbBahbqLj | 3lejckzNYS | MlgnC1JEJn |
| s9Fptw806W | 3lejckzNYS | XrPPnXrnJ3 |
| umALkT56Xy | 3lejckzNYS | grkjTJPNEP |
| 4hGbj2aBVR | 4MVbu2iI15 | x0XpZD2q1n |
| 4xp0ETTkc8 | 4MVbu2iI15 | aI8k5G0Fqi |
+------------+------------+------------+
10 rows in set (0.00 sec)

mysql> select * from contains_ limit 10;
+------------+-------------+--------+
| setID      | modelNumber | count_ |
+------------+-------------+--------+
| 0naFKFb6se | hv5bmT6oG9  |      4 |
```

```
| 0naFKFb6se | JnxJCLw8rc  |      2 |
| 0naFKFb6se | msWH7nqeNv  |      3 |
| 0naFKFb6se | RfBTVUqoof  |      3 |
| 0naFKFb6se | UhCgTjSXvT  |      2 |
| 0nZQJHrEtZ | H7AK7JFI2z  |      5 |
| 0nZQJHrEtZ | kphIGiILcS  |      6 |
| 0nZQJHrEtZ | lqKAZ0fWr5  |      3 |
| 0nZQJHrEtZ | OLFH7SPI6S  |      3 |
| 0nZQJHrEtZ | ouZNXv6e6y  |      5 |
+------------+-------------+--------+
10 rows in set (0.00 sec)

mysql> select * from describes limit 10;
+-------------+------------+
| modelNumber | sku        |
+-------------+------------+
| 00aWzohu8g  | g3v6PqXumq |
| 02SVZ28q47  | KRp6uvT5Nf |
| 0aBIk36Hs2  | TBraGAJvxH |
| 0JCfsD5Pck  | 4FsRw9nHzb |
| 0KSfx7M1IN  | GeGjTFxfw7 |
| 0KSjIO5LDu  | Xrrwxdca1D |
| 0mp9PGPdKw  | fa5G2jkkGG |
| 0T2PaDHCEw  | HIz1MJXQfO |
| 0uzi2lU3dA  | UDYoo8F309 |
| 0vhyAaLCYg  | PDhd8Kn8rC |
+-------------+------------+
10 rows in set (0.00 sec)

mysql> select * from canOrderFrom limit 10;
+------------+------------+----------+
| centerID   | supplierID | leadTime |
+------------+------------+----------+
| 0gkU23fhbT | 06DeQdaf4X |       28 |
| 0gkU23fhbT | 0OaJOGFGdD |        5 |
| 0gkU23fhbT | 3LnmekGoPa |       28 |
| 0gkU23fhbT | 45AVHG6SDL |       17 |
| 0gkU23fhbT | 5SKHZGJsjV |        1 |
| 0gkU23fhbT | 7Hh3kV9mIX |       29 |
| 0gkU23fhbT | 7n94AZBOkB |       28 |
| 0gkU23fhbT | 9kQZDUILDP |       28 |
| 0gkU23fhbT | dWuUGFOe1b |       12 |
| 0gkU23fhbT | EMLwuAKAEG |       26 |
+------------+------------+----------+
10 rows in set (0.00 sec)
```

```
mysql> select * from stocks limit 10;
+------------+------------+
| centerID   | sku        |
+------------+------------+
| 0gkU23fhbT | 6wdiSd4mc0 |
| 0gkU23fhbT | fB8CTGobFM |
| 0gkU23fhbT | fdUqKPeyDL |
| 0gkU23fhbT | INtjFxGvYo |
| 0gkU23fhbT | mISzbMdNjf |
| 0gkU23fhbT | nsfiUC8eEg |
| 0gkU23fhbT | OMeAaXgdtd |
| 0gkU23fhbT | RzHfSIqXgP |
| 0mslNHYadG | C96OgY6RFU |
| 0mslNHYadG | fSiM5um0Ex |
+------------+------------+
10 rows in set (0.00 sec)

mysql> select * from Chair limit 10;
+------------+--------------+------------+---------+------------+------------+
| sku        | numberOfLegs | hasCushion | hasArms | backHeight | seatHeight |
+------------+--------------+------------+---------+------------+------------+
| 0RzRFaHBm4 |            4 |          0 |       0 |         30 |         37 |
| 171xmqOt46 |            4 |          1 |       1 |         27 |         25 |
| 1mtTrTYoiN |            4 |          0 |       1 |         25 |         42 |
| 1OHH0exARc |            5 |          0 |       1 |         32 |         43 |
| 1xIdirGiRI |            5 |          0 |       1 |         25 |         43 |
| 2DUG1Jtxtm |            5 |          0 |       0 |         26 |         32 |
| 2oXo7jOoTO |            4 |          0 |       1 |         29 |         43 |
| 2PijUpIwmL |            4 |          0 |       0 |         27 |         26 |
| 30B5Vsjt9F |            4 |          1 |       0 |         25 |         41 |
| 3c4v9htJja |            4 |          1 |       0 |         25 |         25 |
+------------+--------------+------------+---------+------------+------------+
10 rows in set (0.00 sec)

mysql> select * from Table_ limit 10;
+------------+--------------+---------------+-------------+
| sku        | numberOfLegs | numberOfSeats | shape       |
+------------+--------------+---------------+-------------+
| 0EKsH99oc8 |            8 |             6 | Round       |
| 1masnS2t5q |            7 |             4 | Rectangular |
| 2h36eG3k1r |            7 |             9 | Rectangular |
| 2IQOLQOPsC |            6 |             4 | Round       |
| 2k6FCNLgQa |            9 |             6 | Round       |
| 2orEiGE7st |            5 |            10 | Rectangular |
```

```
| 2sZn2Q9ZtV |            6 |             6 | Rectangular |
| 4arqe70auT |            4 |             9 | Round       |
| 4NeTvOLwsz |            7 |             8 | Rectangular |
| 4OsyaTUMoa |            9 |             5 | Round       |
+------------+--------------+---------------+-------------+
10 rows in set (0.00 sec)

mysql> select * from Desk limit 10;
+------------+-------+-----------------+
| sku        | angle | numberOfDrawers |
+------------+-------+-----------------+
| 0t5AqpGvtk |     7 |               6 |
| 0ZtpDsH7UG |     8 |               1 |
| 1Lnhya8ugI |    11 |               3 |
| 29ilUDyZOJ |    10 |               4 |
| 301ctu2OrC |    34 |               6 |
| 38JWLVLBbI |    14 |               6 |
| 3deDlxVPvv |     9 |               4 |
| 3XZONY6DCR |    39 |               3 |
| 5MugiPD2JZ |     2 |               1 |
| 5sX4ygONIN |    12 |               2 |
+------------+-------+-----------------+
10 rows in set (0.00 sec)

mysql> select * from Stool limit 10;
+------------+--------------+------------+-----------+
| sku        | numberOfLegs | hasCushion | hasSwivel |
+------------+--------------+------------+-----------+
| 0EbNx7hvlO |            3 |          1 |         1 |
| 0snow42N9H |            5 |          0 |         0 |
| 1cJboIO6HA |            4 |          1 |         0 |
| 1IemqAoqLk |            3 |          0 |         1 |
| 1ZIidFPuSJ |            4 |          1 |         0 |
| 2O8vTBpTSp |            4 |          1 |         1 |
| 3CyrMBvrG2 |            5 |          1 |         1 |
| 3qI7CVoPFi |            3 |          1 |         0 |
| 4mFkBSA9sZ |            3 |          0 |         1 |
| 52CLg2zVKj |            3 |          0 |         0 |
+------------+--------------+------------+-----------+
10 rows in set (0.00 sec)

mysql> select * from Cabinet limit 10;
+------------+----------------------+----------+
| sku        | numberOfCompartments | capacity |
+------------+----------------------+----------+
```

```
| 09S4mRvuGE |                    6 | 8        |
| 0eLPBgmxXc |                    3 | 6        |
| 0F6T8XT2R1 |                    1 | 4        |
| 17ApLkPoU6 |                    2 | 9        |
| 1Ddecxm7cb |                    1 | 7        |
| 1MeDZxrjEP |                    4 | 6        |
| 1ykxqSZlb0 |                    1 | 2        |
| 2YXJ88JhZg |                    2 | 3        |
| 3fPhRkO5BI |                    3 | 10       |
| 3jPPB5DgV3 |                    2 | 4        |
+------------+----------------------+----------+
10 rows in set (0.00 sec)

mysql> select * from Bedframe limit 10;
+------------+-----------------+--------+
| sku        | size_           | depth_ |
+------------+-----------------+--------+
| 0EbNx7hvlO | Twin XL         |     35 |
| 1xIdirGiRI | California King |     33 |
| 2oXo7jOoTO | California King |     26 |
| 4mFkBSA9sZ | King            |     41 |
| 4OsyaTUMoa | California King |     38 |
| 4SxmLVmwvL | King            |     24 |
| 4uEa0JgHI2 | Twin            |     25 |
| 4uSXR0shjD | Full            |     28 |
| 4V4wXGK9BV | California King |     31 |
| 53S4nBUolG | Queen           |     34 |
+------------+-----------------+--------+
10 rows in set (0.00 sec)

mysql> select * from features_Feature limit 10;
+-------------+---------------------+--------+
| modelNumber | description         | count_ |
+-------------+---------------------+--------+
| 00aWzohu8g  | Carved Inlays       |      5 |
| 00aWzohu8g  | Fancy Molding       |      3 |
| 00aWzohu8g  | Gold Hinges         |      2 |
| 00aWzohu8g  | Gold Inlaid Designs |      1 |
| 00aWzohu8g  | Ivory Handles       |      5 |
| 00aWzohu8g  | Padded Feet         |      1 |
| 00aWzohu8g  | Studded Corners     |      1 |
| 00aWzohu8g  | Textured            |      4 |
| 02SVZ28q47  | Carved Inlays       |      4 |
| 02SVZ28q47  | Claw Feet           |      2 |
+-------------+---------------------+--------+
```

```
10 rows in set (0.00 sec)
```

6 Part 6

6.1 SQL Schemas

```sql
CREATE TABLE Supplier
  (
     supplierID VARCHAR(10) CHARACTER SET ASCII,
     name_      VARCHAR(50) CHARACTER SET utf8mb4 NOT NULL,
     phone      VARCHAR(12) CHARACTER SET ASCII,
     address    VARCHAR(100) CHARACTER SET utf8mb4,
     country    CHAR(2) CHARACTER SET ASCII DEFAULT 'US',
     website    VARCHAR(50) CHARACTER SET utf8mb4,
     PRIMARY KEY(supplierID),
     CHECK (country REGEXP '^[A-Z]{2}$'),
     CHECK (phone REGEXP '^[0-9]{7,12}$')
  );

CREATE TABLE Designer
  (
     designerID  VARCHAR(10) CHARACTER SET ASCII,
     name_       VARCHAR(50) CHARACTER SET utf8mb4 NOT NULL,
     phone       VARCHAR(12) CHARACTER SET ASCII,
     address     VARCHAR(100) CHARACTER SET utf8mb4,
     country     CHAR(2) CHARACTER SET ASCII DEFAULT 'US',
     website     VARCHAR(50) CHARACTER SET utf8mb4,
     designFocus VARCHAR(100) CHARACTER SET utf8mb4,
     PRIMARY KEY(designerID),
     CHECK (country REGEXP '^[A-Z]{2}$'),
     CHECK (phone REGEXP '^[0-9]{7,12}$')
  );

CREATE TABLE Set_
  (
     setID         VARCHAR(10) CHARACTER SET ASCII,
     name_         VARCHAR(50) CHARACTER SET utf8mb4 NOT NULL,
     catalogYear   DECIMAL(4, 0) UNSIGNED,
```

```sql
      catalogNumber BIGINT UNSIGNED ZEROFILL NOT NULL,
      style_        VARCHAR(30) CHARACTER SET utf8mb4,
      PRIMARY KEY(setID)
   );

CREATE TABLE Model
   (
      modelNumber VARCHAR(10) CHARACTER SET ASCII,
      name_       VARCHAR(50) CHARACTER SET utf8mb4 NOT NULL,
      material    VARCHAR(30) CHARACTER SET utf8mb4,
      upholstery  VARCHAR(30) CHARACTER SET utf8mb4,
      durability  VARCHAR(30) CHARACTER SET utf8mb4,
      color       VARCHAR(30) CHARACTER SET utf8mb4,
      PRIMARY KEY(modelNumber)
   );

CREATE TABLE Item
   (
      sku         VARCHAR(10) CHARACTER SET ASCII,
      length_     DOUBLE UNSIGNED,
      width       DOUBLE UNSIGNED,
      height      DOUBLE UNSIGNED,
      condition_  VARCHAR(30) CHARACTER SET utf8mb4,
      weightLimit DOUBLE UNSIGNED,
      PRIMARY KEY(sku),
      CHECK (length_ > 0.0),
      CHECK (width > 0.0),
      CHECK (height > 0.0),
      CHECK (weightLimit > 0.0)
   );

CREATE TABLE DistributionCenter
   (
      centerID VARCHAR(10) CHARACTER SET ASCII,
      name_    VARCHAR(50) CHARACTER SET utf8mb4 NOT NULL,
      phone    VARCHAR(12) CHARACTER SET ASCII,
      address  VARCHAR(100) CHARACTER SET utf8mb4,
      country  CHAR(2) CHARACTER SET ASCII DEFAULT 'US',
      website  VARCHAR(50) CHARACTER SET utf8mb4,
      PRIMARY KEY(centerID),
      CHECK (country REGEXP '^[A-Z]{2}$'),
      CHECK (phone REGEXP '^[0-9]{7,12}$')
   );

CREATE TABLE make
```

```
  (
     supplierID VARCHAR(10) CHARACTER SET ASCII,
     designerID VARCHAR(10) CHARACTER SET ASCII,
     setID      VARCHAR(10) CHARACTER SET ASCII,
     PRIMARY KEY(supplierID, designerID, setID),
     FOREIGN KEY(supplierID) REFERENCES Supplier(supplierID),
     FOREIGN KEY(designerID) REFERENCES Designer(designerID),
     FOREIGN KEY(setID) REFERENCES Set_(setID)
  );

CREATE TABLE contains_
  (
     setID       VARCHAR(10) CHARACTER SET ASCII,
     modelNumber VARCHAR(10) CHARACTER SET ASCII,
     count_      TINYINT UNSIGNED DEFAULT 1,
     PRIMARY KEY(setID, modelNumber),
     FOREIGN KEY(setID) REFERENCES Set_(setID),
     FOREIGN KEY(modelNumber) REFERENCES Model(modelNumber),
     CHECK (count_ > 0)
  );

CREATE TABLE describes
  (
     modelNumber VARCHAR(10) CHARACTER SET ASCII NOT NULL,
     sku         VARCHAR(10) CHARACTER SET ASCII,
     PRIMARY KEY(sku),
     FOREIGN KEY(modelNumber) REFERENCES Model(modelNumber),
     FOREIGN KEY(sku) REFERENCES Item(sku)
  );

CREATE TABLE canOrderFrom
  (
     centerID   VARCHAR(10) CHARACTER SET ASCII,
     supplierID VARCHAR(10) CHARACTER SET ASCII,
     leadTime   DOUBLE UNSIGNED,
     PRIMARY KEY(centerID, supplierID),
     FOREIGN KEY(centerID) REFERENCES DistributionCenter(centerID),
     FOREIGN KEY(supplierID) REFERENCES Supplier(supplierID)
  );

CREATE TABLE stocks
  (
     centerID VARCHAR(10) CHARACTER SET ASCII NOT NULL,
     sku      VARCHAR(10) CHARACTER SET ASCII,
     PRIMARY KEY(sku),
```

```
      FOREIGN KEY(centerID) REFERENCES DistributionCenter(centerID),
      FOREIGN KEY(sku) REFERENCES Item(sku)
   );

CREATE TABLE Chair
   (
      sku          VARCHAR(10) CHARACTER SET ASCII,
      numberOfLegs TINYINT UNSIGNED DEFAULT 4,
      hasCushion   BOOL DEFAULT false,
      hasArms      BOOL,
      backHeight   DOUBLE UNSIGNED,
      seatHeight   DOUBLE UNSIGNED,
      PRIMARY KEY(sku),
      FOREIGN KEY(sku) REFERENCES Item(sku),
      CHECK (numberOfLegs > 0),
      CHECK (backHeight > 0.0),
      CHECK (seatHeight > 0.0)
   );

CREATE TABLE Table_
   (
      sku           VARCHAR(10) CHARACTER SET ASCII,
      numberOfLegs  TINYINT UNSIGNED DEFAULT 4,
      numberOfSeats TINYINT UNSIGNED,
      shape         VARCHAR(30) CHARACTER SET utf8mb4,
      PRIMARY KEY(sku),
      FOREIGN KEY(sku) REFERENCES Item(sku),
      CHECK (numberOfLegs > 0),
      CHECK (numberOfSeats > 0)
   );

CREATE TABLE Desk
   (
      sku             VARCHAR(10) CHARACTER SET ASCII,
      angle           DOUBLE DEFAULT 0.0,
      numberOfDrawers TINYINT UNSIGNED,
      PRIMARY KEY(sku),
      FOREIGN KEY(sku) REFERENCES Item(sku),
      CHECK (angle > -360.0 AND angle < 360.0),
      CHECK (numberOfDrawers > 0)
   );

CREATE TABLE Stool
   (
      sku          VARCHAR(10) CHARACTER SET ASCII,
```

```
    numberOfLegs TINYINT UNSIGNED,
    hasCushion   BOOL,
    hasSwivel    BOOL,
    PRIMARY KEY(sku),
    FOREIGN KEY(sku) REFERENCES Item(sku),
    CHECK (numberOfLegs > 0)
  );

CREATE TABLE Cabinet
  (
    sku                  VARCHAR(10) CHARACTER SET ASCII,
    numberOfCompartments TINYINT UNSIGNED,
    capacity             VARCHAR(30) CHARACTER SET utf8mb4,
    PRIMARY KEY(sku),
    FOREIGN KEY(sku) REFERENCES Item(sku),
    CHECK (numberOfCompartments > 0)
  );

CREATE TABLE Bedframe
  (
    sku    VARCHAR(10) CHARACTER SET ASCII,
    size_  VARCHAR(30) CHARACTER SET utf8mb4,
    depth_ DOUBLE,
    PRIMARY KEY(sku),
    FOREIGN KEY(sku) REFERENCES Item(sku)
  );

CREATE TABLE features_Feature
  (
    modelNumber VARCHAR(10) CHARACTER SET ASCII,
    description VARCHAR(50) CHARACTER SET utf8mb4,
    count_      TINYINT UNSIGNED DEFAULT 1,
    PRIMARY KEY(modelNumber, description),
    FOREIGN KEY(modelNumber) REFERENCES Model(modelNumber),
    CHECK (count_ > 0)
  );
```

6.2 Sample Queries

(A) **Intent:** "How many models has each designer designed?"

Query:

```
SELECT M.designerID,
       COUNT(C.modelNumber)
FROM   Designer D
       LEFT OUTER JOIN make M
```

```
                    ON ( D.designerID = M.designerID )
        INNER JOIN contains_ C
                ON ( M.setID = C.setID )
GROUP  BY M.designerID;
```

Result:

```
+------------+---------------------+
| designerID | count(C.modelNumber) |
+------------+---------------------+
| 3lejckzNYS |                  69 |
| 4MVbu2iI15 |                 170 |
| 7RYpyw9es0 |                 212 |
| 7ZZbCsXnv0 |                 210 |
| A7oA1v9Ax1 |                 212 |
| AZadqlHsUN |                 232 |
| b9yOGUx3pl |                 213 |
| bBMMbiXWX2 |                 189 |
| bDcoTRYgku |                 148 |
| BejwYSNzm7 |                 142 |
| bjSmt6EX8o |                 193 |
| bnQDB9V4ZQ |                 215 |
| bOQ84LG8yQ |                 246 |
| BVP4o4g0u6 |                 121 |
| c1jIajAyha |                 142 |
| czfBYIFNhs |                 252 |
| eAm5FaKjru |                 196 |
| ejSgB4P19T |                 215 |
| EZn2c6Sqao |                 164 |
| fLM0vFMD6h |                 190 |
| HACGEeYiTg |                 182 |
| hL6koxT8vK |                 265 |
| HULdxdPYgo |                 130 |
| HUPang5JW4 |                 181 |
| in19yTwFqy |                 254 |
| jMt9cpvHJ8 |                 218 |
| kFNSGfDIXN |                 200 |
| KH4hmznKQN |                 252 |
| KoaWPsykpt |                 244 |
| L2vFnWn5yt |                 170 |
| lA2l4dUPAN |                 265 |
| MBxLmTyDx0 |                 152 |
| nUFwyC0BAj |                 167 |
| nxtwKjphvt |                 260 |
| OLkEycvtOv |                  99 |
| p3f6twELII |                 203 |
| P9Vg4XQ5AK |                 267 |
```

```
| pIdz2ArSJd |                  168 |
| PWFixIVSN0 |                  197 |
| ql0bQqFgKx |                  224 |
| rdYk2JSFOZ |                  229 |
| sUUNXUZjSB |                  174 |
| TnFL7eVZD9 |                  246 |
| UKHmNCJ1Ep |                  211 |
| uQJBa7yRRm |                  240 |
| VXBvjILW4l |                  255 |
| VyLXDToji5 |                  186 |
| Xi0fOUOila |                  200 |
| YOpJVyms0T |                  151 |
| yz0xcnsOMA |                  212 |
+------------+----------------------+
50 rows in set (0.01 sec)
```

Explanation: This query is plausible because someone might want to check which designers were more prolific; this might inform their decision on whose furniture to buy.

This result is sensible, though somewhat unlikely in the real world; there are 50 designers and 1000 models, and each designer has designed about 200 models on average, so each model has about 10 designers. Further investigation into the data validated this calculation, so the query gave the intended result, even if that result was unintuitive.

(*B*) **Intent:** "What is the average lead time from suppliers to distribution centers when both are in the same country?"

Query:

```
SELECT AVG(leadTime)
FROM   Supplier S,
       DistributionCenter C,
       canOrderFrom O
WHERE  S.supplierID = O.supplierID
   AND C.centerID = O.centerID
   AND S.country = C.country;
```

Result:

```
+-------------------+
| AVG(leadTime)     |
+-------------------+
| 5.333333333333333 |
+-------------------+
1 row in set (0.00 sec)
```

Explanation: This query is plausible because someone might want to know how long, on average, they'd have to wait for furniture to arrive at the distribution centers after being ordered.

This result is sensible because it gives a single result to a query that contains only an aggregation function and no grouping.

(C) **Intent:** "Which suppliers can get me a claw-footed bedframe in less than a week?"

Query:
```
SELECT DISTINCT O.supplierID
FROM   Bedframe B,
       describes D,
       features_Feature F,
       canOrderFrom O,
       stocks S
WHERE  B.sku = D.sku
   AND D.modelNumber = F.modelNumber
   AND F.description = 'Claw Feet'
   AND O.centerID = S.centerID
   AND S.sku = B.sku
   AND O.leadTime < 7.0;
```

Result:
```
+------------+
| supplierID |
+------------+
| 45AVHG6SDL |
| dWuUGFOe1b |
| lGSxEQQ8bN |
| 4hGbj2aBVR |
| socFXUsXAD |
| 0x1V8UVLqh |
| umALkT56Xy |
| xUMVYgBI7J |
| FE34LTqb2p |
| ldbBahbqLj |
| ryNb8O1TRs |
| tRJBEnEFaO |
| ATeybFIuTQ |
| 7n94AZBOkB |
| Op8rarVKxa |
| FyAzhOpNly |
| WY3AmkQmxs |
| xyEInQvE1U |
| ZxsJeaFpg5 |
| 06DeQdaf4X |
| 5SKHZGJsjV |
| FVzpxLJFKK |
| uluou8izCd |
| nZ4msLXxTY |
| x7FolH1EsA |
| xJZiBD5DIo |
| cUrmX3GV4D |
| KQIGJ9eDk5 |
```

```
| 8pYnVWzpzR |
| 9Hzdyajzdu |
| 9kQZDUILDP |
| 3LnmekGoPa |
| 4xp0ETTkc8 |
| IBvWjkFney |
| XIFO3VhtQH |
+------------+
35 rows in set (0.00 sec)
```

Explanation: This query is plausible because a customer might very well want a bedframe—with particular attributes, even—within a specified time limit.

This result seems reasonable, because it feels fairly realistic that claw-footed bedframes wouldn't be particularly rare, so 35 of our 50 available suppliers carrying them seems fine. It should be noted that, the first time this query was run, we forgot the `DISTINCT` keyword after `SELECT`; this gave us 100 (obviously non-unique) results, and taught us a lesson about not assuming automatic distinctness in the result of a complex query just because the result column is a primary key in its original table.

(*D*) **Intent:** "How many items is my old buddy Harold stocking in that warehouse of his?"

Query:
```
SELECT COUNT(*)
FROM   stocks S,
       DistributionCenter D
WHERE  S.centerID = D.centerID
   AND D.name_ = 'Harold and His Big Ass Warehouse';
```

Result:
```
+----------+
| COUNT(*) |
+----------+
|        7 |
+----------+
1 row in set (0.00 sec)
```

Explanation: This query is plausible because somebody might actually care about Harold's business success (or lack thereof). We wanted to make a query that involved a specific name or attribute to make sure we could return small data as well as larger data.

Apparently Harold has a lot of wasted space right now; blame the economy.

(*E*) **Intent:** "Which sets have more than two chairs and at least one table?"

Query:
```
SELECT DISTINCT C.setID
FROM   contains_ C
WHERE  (SELECT COUNT(DISTINCT I.sku) * C.count_
```

```
        FROM   Table_ I,
               describes D
        WHERE  D.sku = I.sku
           AND D.modelNumber = C.modelNumber) >= 1
    AND (SELECT COUNT(DISTINCT I.sku) * C.count_
        FROM   Chair I,
               describes D
        WHERE  D.sku = I.sku
           AND D.modelNumber = C.modelNumber) > 2;
```

Result: `Empty set (0.00 sec)`

Explanation: This query is plausible because someone might want some dinner furniture in a professionally matched set.

I guess we don't carry any of that kind of set; oh well!

7 Part 7

7.1 Extra Functionality

For extra functionality, we decided to implement a search function into the webpage. This search bar allows a “customer” to search the database in a “keyword”-based fashion, just like they would on a real store website This way, we can search the items that our store can sell in a way that an actual customer would and return results that are relevant to that search. For example, searching for “Table” returns all of the tables in the database and even shows a picture next to each item. Granted, this is just a placeholder image for each of the different items that the store sells but it is a proof of concept more than anything.

7.2 Domain Usability

Someone working in a furniture store might reasonably consider using our database. Our web enabled database can search for specific pieces of furniture that customers would search for, and this search is done in a similar way to other product websites. The data is also presented in a list that doesn’t mirror the style of an actual product page but presents the appropriate data. This is enough of a proof of concept that it could present it in a way that “looks good” to a customer.

Our approach is largely worse than similar systems. This is simply due to the style in which the data from the database is presented. With ample time, we are confident that we could ensure that the data for each tuple is presented in a usable way that could mirror a product page. Our approach to the presentation of the data is what we believe to be close to how professional databases store information on products in an online store. Our presentation, however, leaves much to be desired, as simple tables with almost arbitrary information are not aesthetically pleasing.

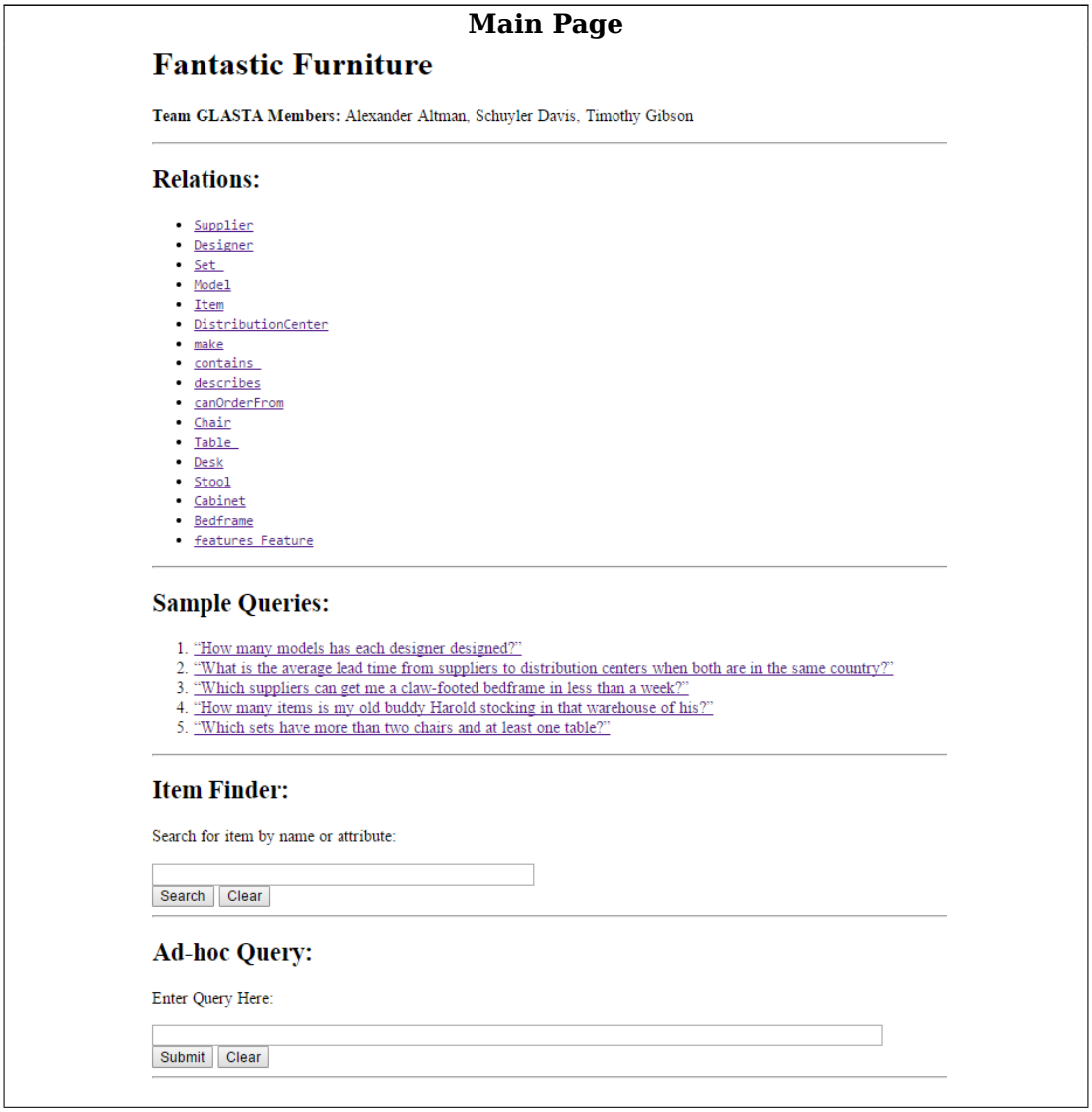
7.3 SQL Injection Security

On line 38 of `query.php`, there is an assertion that the query about to be run does not contain the full words `CREATE`, `ALTER`, `DROP`, or `RENAME` in any casing. On line 3 of `conn.php`, the PHP environment is instructed to bail immediately upon any assertions failing. On line 9 of `conn.php`, the PDO engine is instructed to throw an exception upon any failing operation.

These three steps, taken together, ensure that no `CREATE`, `ALTER`, `DROP`, or `RENAME` statements are ever executed via `query.php`.

In addition, on lines 39–45 of `finder.php`, PDO's facility for safely binding variables to query parameters via prepared statements is used to allow arbitrary search text to be included into a query without risking SQL injection.

7.4 Website Screenshots



Relation View

supplierID [VAR_STRING]	name_ [VAR_STRING]	phone [VAR_STRING]	address [VAR_STRING]	country [STRING]	website [VAR_STRING]
06DeQdaf4X	The Snowy Pencil Leather Company	8039090134	379 Virginia St. Niagara Falls, NY 14304	TR	www.TheSnowyPencilLeatherCompany.biz
0OaJOGFGdD	The Cloudy Chicken Fabrics Company	8749357327	8247 S. Hamilton Drive Cedar Rapids, IA 52402	TO	www.TheCloudyChickenFabricsCompany.info
0x1V8UVLqh	The Deep Lamp Lumber Company	7463153612	72 Somerset Lane Ypsilanti, MI 48197	AQ	www.TheDeepLampLumberCompany.info
3H7BBv3j7C	Tall Mouse Supplier	4123072387	9444 Blackburn Lane Wenatchee, WA 98801	PA	www.TallMouseSupplier.org
3LnmekGoPa	The Happy Chinchilla Metal Company	9115174479	50 Catherine Lane Huntsville, AL 35803	ZM	www.TheHappyChinchillaMetalCompany.com
45AVHG6SDL	Big Light-Switch Metal	2219847475	9916 Bridgeton Ave. Austin, MN 55912	AQ	www.BigLight-SwitchMetal.org
4hGbj2aBVR	Red Baboon Metal	8433984003	860 Glen Ridge Rd. Whitestone, NY 11357	BT	www.RedBaboonMetal.net
4xp0ETTkc8	The White Skunk Metal Company	2695261566	52 Deerfield Lane Woodhaven, NY 11421	DJ	www.TheWhiteSkunkMetalCompany.info
5Hq7fF9aK0	The Opaque Pear Supplier Company	9201886129	7254 Hickory Ave. Centreville, VA 20120	CI	www.TheOpaquePearSupplierCompany.org
5SKHZGJsjV	Transparent Cherry Lumber	4623416776	1 Redwood Rd. Roanoke, VA 24012	ZW	www.TransparentCherryLumber.org
7Hh3kV9mIX	Little Turtle Metal	1858449978	7377 Orchard Dr. Perrysburg, OH 43551	FR	www.LittleTurtleMetal.org
7n94AZBOkB	The Freezing Fan Supplier Company	5834647945	190 North Franklin St. West Chester, PA 19380	PR	www.TheFreezingFanSupplierCompany.biz
8pYnVWzpzR	The Peaceful Tomato Supplier Company	3664563759	7 W. Fairview Road Mechanicsburg, PA 17050	GW	www.ThePeacefulTomatoSupplierCompany.biz
9Hzdyajzdu	Thin Fan Logging	9578045441	8 Hillcrest St. Buford, GA 30518	LU	www.ThinFanLogging.info
9kQZDUILDP	The Red Fish Logging Company	1847809810	95 Winchester Ave. Casselberry, FL 32707	LK	www.TheRedFishLoggingCompany.info
ATeybFIuTQ	Stupid Rabbit Lumber	2589423308	7914 Washington Dr. Harrison Township, MI 48045	US	www.StupidRabbitLumber.com
cUrmX3GV4D	Thin Beaver Metal	7435961046	9433 Hickory Street Tacoma, WA 98444	MQ	www.ThinBeaverMetal.info
DIq94L7rPh	Deep Elephant Supplier	2127952315	44 Hilltop Street Rolla, MO 65401	CG	www.DeepElephantSupplier.net
dWuUGFOe1b	Small Rabbit Logging	7247218336	8908 Pilgrim Lane Oviedo, FL 32765	IQ	www.SmallRabbitLogging.com
EMLwuAKAEG	Blue Pear Liquidators	9921926501	437 Elizabeth St. Apopka, FL 32703	YE	www.BluePearLiquidators.org
EOnkzmV7ND	The Simple Robot Fabrics Company	3872132793	8266 East Hillside Ave. Saint Louis, MO 63109	SG	www.TheSimpleRobotFabricsCompany.info
FE34LTqb2p	The Tiny Tomato Lumber Company	8743091544	536 Pulaski Street Wakefield, MA 01880	SN	www.TheTinyTomatoLumberCompany.org
FVzpxLJFKK	Small Turkey Supplier	3293196822	276 Wood Road Villa Rica, GA 30180	AE	www.SmallTurkeySupplier.net
FyAzhOpNly	The Sad Beaver Liquidators Company	8636586436	8296 North Lane Shrewsbury, MA 01545	NU	www.TheSadBeaverLiquidatorsCompany.info
IBvWjkFney	The Brown Fan Logging Company	5911329769	937 Cypress Street Butte, MT 59701	SX	www.TheBrownFanLoggingCompany.com
KQIGJ9eDk5	The Brown Elephant Liquidators Company	5501650439	486 Sugar St. Park Forest, IL 60466	KE	www.TheBrownElephantLiquidatorsCompany.net
ldbBahbqLj	Foggy Skunk Lumber	8485622783	9810 Edgefield St. Natick, MA 01760	CG	www.FoggySkunkLumber.com
lGSxEQQ8bN	The Fat Light-Switch Lumber Company	7003923806	99 Lower River Drive Mount Vernon, NY 10550	CO	www.TheFatLight-SwitchLumberCompany.info
nZ4msLXxTY	Peaceful Fox Supplier	5416771400	9091 Old Marshall Rd. Bethel Park, PA 15102	GP	www.PeacefulFoxSupplier.com
Op8rarVKxa	The Tiger Lumber Company	1891253286	18 Sheffield Ave. Butte, MT 59701	MQ	www.TheTigerLumberCompany.net
QGoBUjHkki	Thin Apple Metal	7492027753	8414 Oxford Ave. Hixson, TN 37343	RE	www.ThinAppleMetal.com
ryNb8O1TRs	The Foggy Pigeon Supplier Company	7421854946	8288 Oakland Lane Cambridge, MA 02138	SY	www.TheFoggyPigeonSupplierCompany.com
s56xoRSQrg	The Tiny Camel Lumber Company	1355079530	600 Pennsylvania St. Deland, FL 32720	AZ	www.TheTinyCamelLumberCompany.biz
s9Fptw806W	The Intelligent Scarf Liquidators Company	6712497760	347 Wakehurst Street Rahway, NJ 07065	CO	www.TheIntelligentScarfLiquidatorsCompany.info
socFXUsXAD	The Tiny Gecko Supplier Company	4876178883	3 Ridge Court Maryville, TN 37803	VN	www.TheTinyGeckoSupplierCompany.com
tRJBEnEFaO	The Cheerful Skunk Logging Company	9195602302	8223 Marconi St. Dawsonville, GA 30534	DO	www.TheCheerfulSkunkLoggingCompany.net
UhCJbQG0jF	The Cold Pigeon Leather Company	8135651978	8248 River Ave. Falls Church, VA 22041	CR	www.TheColdPigeonLeatherCompany.info
uluou8izCd	The Stupid Alligator Supplier Company	8511426114	69 Bay Meadows Lane Bridgeton, NJ 08302	TJ	www.TheStupidAlligatorSupplierCompany.info
umALkT56Xy	The Thin Cat Leather Company	1046485315	92 W. Princeton Rd. Long Beach, NY 11561	CF	www.TheThinCatLeatherCompany.org
VQtT65VuGn	Acute Banana Liquidators	7034816719	488 Center Lane Saint Albans, NY 11412	GH	www.AcuteBananaLiquidators.org
wU1wdmKEca	The Orange Donkey Fabrics Company	4249980563	30 Lawrence Street North Ridgeville, OH 44039	ER	www.TheOrangeDonkeyFabricsCompany.org
WY3AmkQmxs	The Mouse Metal Company	8468272349	166 South Glenlake Dr. Garner, NC 27529	MQ	www.TheMouseMetalCompany.com
x7FoIH1EsA	Stormy Snake Lumber	3012684277	554 Fairfield St. Torrington, CT 06790	GS	www.StormySnakeLumber.net
XIFO3VhtQH	Little Ram Fabrics	8744685507	8124 North Belmont Dr. Kalamazoo, MI 49009	QA	www.LittleRamFabrics.biz
xJZiBD5DIo	The Tall Fork Lumber Company	2738728093	7083 Green Court Pembroke Pines, FL 33028	JM	www.TheTallForkLumberCompany.net
xSCCF3BYWn	Fat Zebra Leather	7071474222	9499 Manchester Rd. Fargo, ND 58102	AL	www.FatZebraLeather.org
xUMVYgB17J	The Angry Pen Fabrics Company	4937283463	8942 Myrtle St. New Port Richey, FL 34653	GL	www.TheAngryPenFabricsCompany.net
xyEInQvE1U	The Cheeky Banana Leather Company	4204843909	7583 N. Howard Road Butte, MT 59701	VG	www.TheCheekyBananaLeatherCompany.net
ZcXBxuYu7W	The Plain Beaver Metal Company	1437523438	827 Dogwood Ave. Saginaw, MI 48601	ZW	www.ThePlainBeaverMetalCompany.org
ZxsJeaFpg5	Plain Fan Metal	8256624098	904 Creekside St. Pottstown, PA 19464	LC	www.PlainFanMetal.info

Sample Query

Query: `SELECT AVG(leadTime) FROM Supplier S, DistributionCenter C, canOrderFrom O WHERE S.supplierID = O.supplierID AND C.centerID = O.centerID AND S.country = C.country;`

AVG(leadTime) [DOUBLE]
5.333333333333333

Search Query

Image	sku [VAR_STRING]	modelNumber [VAR_STRING]	name_ [VAR_STRING]	material [VAR_STRING]	upholstery [VAR_STRING]	durability [VAR_STRING]	color [VAR_STRING]	length_ [DOUBLE]	width [DOUBLE]	height [DOUBLE]	condition_ [VAR_STRING]	weightLimit [DOUBLE]
	Qp3AiGFhUp	2AOOGS7BoV	Stylish Gaunt Chair	Hickory	Nylon	Somewhat Sturdy	Blue	68	65	70	New	357
	V6C78UkTWy	5A75nuYO0H	Homely Gaunt Chair	Cottonwood	Leather	Somewhat Sturdy	Sky Blue	67	57	59	New	424
	eYzu2ezn4s	78rJKTUJXk	Large Gaunt Chair	Afzelia	Leather	Indestructable	Pink	63	59	53	New	343
	Di0tKcGWvi	EHOc6EXxFj	Timeless Gaunt Chair	Alder	Vinyl	Very Sturdy	Olive	62	69	49	New	321
	miLYw1bfaU	hlv0i3VKBJ	Stylish Gaunt Chair	Balsa	Cotton Blend	Very Strong	Salmon	69	62	49	New	381
	GygfE2IPwp	itcwxz8x5v	Eccentric Gaunt Chair	Pine	Acetate	Wobbly	Turquoise	59	52	61	New	446
	NoemC8Mkpg	lallwicfBB	Strong Gaunt Chair	Afzelia	Cotton Blend	Very Sturdy	Magenta	62	71	65	New	318
	PTvlI5lnj5	MH7TbchPTR	Modern Gaunt Chair	Afzelia	Cotton Blend	Somewhat Wobbly	White	71	69	54	New	499
	neFwVZmrN4	MMz1JGlIKb	Indian Gaunt Chair	Hickory	Silk	Indestructable	Blue	69	55	56	New	308
	qHCsnLF5Pe	ouZNXv6e6y	Modern Gaunt Chair	Steel	Wool	Somewhat Sturdy	Indigo	63	51	69	New	308
	SdxFEZ4vmm	qxl6oMfZru	Large Gaunt Chair	Purpleheart	Silk	Somewhat Sturdy	Grey	70	61	52	New	451
	0F6T8XT2R1	vK4Rp98p4A	Futuristic Gaunt Chair	Lindens	Vinyl	Very Wobbly	Olive	50	59	53	New	310
	u2HcbBBPYu	VT0XZUpVfr	Contemporary Gaunt Chair	Steel	Linen	Very Wobbly	Salmon	67	59	62	New	376
	F2a7ksM6lh	XlxqxjH0In	Modern Gaunt Chair	Maple	Olefin	Sturdy	Teal	48	51	49	New	326

Ad-Hoc Query

Query: `SELECT * FROM Designer WHERE designFocus LIKE "Transitional"`

designerID [VAR_STRING]	name_ [VAR_STRING]	phone [VAR_STRING]	address [VAR_STRING]	country [STRING]	website [VAR_STRING]	designFocus [VAR_STRING]
hL6koxT8vK	Mazie Frye	5501650439	741 Country Dr. Ypsilanti, MI 48197	AM	www.MazieFrye.info	Transitional
lA2l4dUPAN	Lakeshia Flowers	5911329769	9635 West St Margarets Ave. Richmond Hill, NY 11418	GF	www.LakeshiaFlowers.biz	Transitional
MBxLmTyDx0	Jere Delacruz	7492027753	607 Garden Rd. Pataskala, OH 43062	AR	www.JereDelacruz.org	Transitional
UKHmNCJ1Ep	Roxane Parrish	8135651978	64 N. Annadale Ave. Raeford, NC 28376	VC	www.RoxaneParrish.info	Transitional

Group Work

Timothy: Timothy was the second draft coder, scripter, and art director. Timothy checked Alexander's work, making sure the database implementation was coherent. Timothy also wrote the bulk of the scripts that generated the data for the database. Timothy also worked alongside Alexander in making sure the implementation was on task. Finally, Timothy produced any non-code, non-diagrammatic assets that the various project stages needed.

Alexander: Alexander was the dedicated first draft coder for all project parts. He made sure that the basic framework of the database was always at least minimally functional, and had some input on later design stages as well.

Schuyler: Schuyler filled the role of supervisory and conceptualizing. Schuyler was in charge of stepping back and examining project implementation from different angles. He also ensured things "made sense" for our data domain. He filled a mostly "administrative" role.

Appendix: Website Source Code and Resources

Source Code

Note that the contents of the file `info.php` is not included here for security reasons; it contains only the definitions of the global variables `username` and `passwork`. The misspelling of the latter variable's name was not intentional, but it has been maintained anyway for backwards compatibility purposes.

index.php

```
<!-- -*-html-*- -->
<!DOCTYPE html>
<html lang="en-US">

<head>
    <title>Fantastic Furniture</title>
</head>

<body>

    <h1>Fantastic Furniture</h1>
    <p><strong>Team GLASTA Members:</strong> Alexander Altman, Schuyler Davis,
     ↪  Timothy Gibson</p>

    <hr>
    <h2>Relations:</h2>

    <ul>
        <li>
            <a href="relation.php?relation=Supplier"><code>Supplier</code></a>
        </li>
        <li>
            <a href="relation.php?relation=Designer"><code>Designer</code></a>
```

```
        </li>
        <li>
            <a href="relation.php?relation=Set_"><code>Set_</code></a>
        </li>
        <li>
            <a href="relation.php?relation=Model"><code>Model</code></a>
        </li>
        <li>
            <a href="relation.php?relation=Item"><code>Item</code></a>
        </li>
        <li>
            <a href="relation.php?relation=DistributionCenter"><code>DistributionC
            enter</code></a>
        </li>
        <li>
            <a href="relation.php?relation=make"><code>make</code></a>
        </li>
        <li>
            <a href="relation.php?relation=contains_"><code>contains_</code></a>
        </li>
        <li>
            <a href="relation.php?relation=describes"><code>describes</code></a>
        </li>
        <li>
            <a href="relation.php?relation=canOrderFrom"><code>canOrderFrom</code>
            </a>
        </li>
        <li>
            <a href="relation.php?relation=Chair"><code>Chair</code></a>
        </li>
        <li>
            <a href="relation.php?relation=Table_"><code>Table_</code></a>
        </li>
        <li>
            <a href="relation.php?relation=Desk"><code>Desk</code></a>
        </li>
        <li>
            <a href="relation.php?relation=Stool"><code>Stool</code></a>
        </li>
        <li>
            <a href="relation.php?relation=Cabinet"><code>Cabinet</code></a>
        </li>
        <li>
            <a href="relation.php?relation=Bedframe"><code>Bedframe</code></a>
        </li>
```

```
        <li>
            <a href="relation.php?relation=features_Feature"><code>features_Featur
            ↪  e</code></a>
        </li>
    </ul>

    <hr>
    <h2>Sample Queries:</h2>

    <ol>
        <li>
            <a href="query.php?query=1">&ldquo;How many models has each designer
            ↪  designed?&rdquo;</a>
        </li>
        <li>
            <a href="query.php?query=2">&ldquo;What is the average lead time from
            ↪  suppliers to distribution centers when both are in the same
            ↪  country?&rdquo;</a>
        </li>
        <li>
            <a href="query.php?query=3">&ldquo;Which suppliers can get me a
            ↪  claw-footed bedframe in less than a week?&rdquo;</a>
        </li>
        <li>
            <a href="query.php?query=4">&ldquo;How many items is my old buddy
            ↪  Harold stocking in that warehouse of his?&rdquo;</a>
        </li>
        <li>
            <a href="query.php?query=5">&ldquo;Which sets have more than two
            ↪  chairs and at least one table?&rdquo;</a>
        </li>
    </ol>

    <hr>
    <h2>Item Finder:</h2>

    <form action="finder.php" method="post">
        <p>Search for item by name or attribute:</p>
        <input type="text" name="searchText" value="" size="50">
        <br>
        <input type="submit" value="Search">
        <input type="reset" value="Clear">
    </form>

    <hr>
```

```
    <h2>Ad-hoc Query:</h2>

    <form action="query.php?query=6" method="post">
        <p>Enter Query Here:</p>
        <input type="text" name="userQuery" value="" size="100">
        <br>
        <input type="submit" value="Submit">
        <input type="reset" value="Clear">
    </form>

    <hr>

</body>

</html>
```

conn.php

```
<!-- -*-html-*- -->
<?php
assert_options(ASSERT_BAIL, 1);
$servername = "localhost";
$dbname     = "aaltman";
require 'info.php';
try {
    $conn = new PDO("mysql:host=$servername;dbname=$dbname", $username, $passwork);
    $conn->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
}
catch (PDOException $e) {
    echo "<strong>Error:</strong> " . $e->getMessage();
    die;
}
?>
```

relation.php

```
<!-- -*-html-*- -->
<!DOCTYPE html>
<html lang="en-US">

<head>
    <title>Fantastic Furniture</title>
    <style>
        table {
            border-collapse: collapse;
```

```
        }

        table,
        th,
        td {
            border: 1px solid gray;
        }
    </style>
</head>

<body>

    <?php
require 'conn.php';
$validNames   = array(
    "Supplier",
    "Designer",
    "Set_",
    "Model",
    "Item",
    "DistributionCenter",
    "make",
    "contains_",
    "describes",
    "canOrderFrom",
    "Chair",
    "Table_",
    "Desk",
    "Stool",
    "Cabinet",
    "Bedframe",
    "features_Feature"
);
$relationName = $_GET["relation"];
assert(in_array($relationName, $validNames, true), "Invalid relation name!");
$result  = $conn->query("SELECT * FROM $relationName");
$numCols = $result->columnCount();

echo "<table><thead><tr>";
for ($colNum = 0; $colNum < $numCols; $colNum++) {
    $colMeta = $result->getColumnMeta($colNum);
    echo "<th><code>" . $colMeta["name"] . "</code> [<code>" .
     ↪  $colMeta["native_type"] . "</code>]</th>";
}
echo "</tr></thead><tbody>";
```

```
foreach ($result->fetchAll() as $resultRow) {
    echo "<tr>";
    for ($colNum = 0; $colNum < $numCols; $colNum++) {
        echo "<td>" . $resultRow[$colNum] . "</td>";
    }
    echo "</tr>";
}
echo "</tbody></table>";

$result->closeCursor();
$result = null;
$conn   = null;
            ?>

</body>

</html>
```

finder.php

```
<!-- -*-html-*- -->
<!DOCTYPE html>
<html lang="en-US">

<head>
    <title>Fantastic Furniture</title>
    <style>
        table {
            border-collapse: collapse;
        }

        table,
        th,
        td {
            border: 1px solid gray;
        }

        img {
            max-width: 100%;
            height: auto;
        }
    </style>
    <?php
        // copied with gratitude from http://stackoverflow.com/a/619725/1133298
        function endswith($string, $test) {
```

```
            $strlen = strlen($string);
            $testlen = strlen($test);
            if ($testlen > $strlen) return false;
            return substr_compare($string, $test, $strlen - $testlen, $testlen) === 0;
        }
    ?>
</head>

<body>

    <?php
require 'conn.php';
$searchText = $_POST["searchText"];
$result  = $conn->prepare("SELECT * FROM (SELECT DISTINCT * FROM Model M join describes D using (modelNumber) join Item I using (sku) where M.name_ LIKE :searchText1 OR M.material LIKE :searchText2 OR M.upholstery LIKE :searchText3 OR M.durability LIKE :searchText4 OR M.color LIKE :searchText5 OR I.condition_ LIKE  :searchText6) R ORDER BY R.modelNumber, R.sku;");
$result->bindValue(":searchText1", "%{$searchText}%", PDO::PARAM_STR);
$result->bindValue(":searchText2", "%{$searchText}%", PDO::PARAM_STR);
$result->bindValue(":searchText3", "%{$searchText}%", PDO::PARAM_STR);
$result->bindValue(":searchText4", "%{$searchText}%", PDO::PARAM_STR);
$result->bindValue(":searchText5", "%{$searchText}%", PDO::PARAM_STR);
$result->bindValue(":searchText6", "%{$searchText}%", PDO::PARAM_STR);
$result->execute();
$numCols = $result->columnCount();

echo "<table><thead><tr><th>Image</th>";
for ($colNum = 0; $colNum < $numCols; $colNum++) {
    $colMeta = $result->getColumnMeta($colNum);
    echo "<th><code>" . $colMeta["name"] . "</code> [<code>" . $colMeta["native_type"] . "</code>]</th>";
}
echo "</tr></thead><tbody>";
foreach ($result->fetchAll() as $resultRow) {
    if (endswith($resultRow["name_"], "Chair")) {
        $imageTag = '<img src="chair.png" alt="Chair">';
    } else if (endswith($resultRow["name_"], "Cabinet")) {
        $imageTag = '<img src="cabinet.png" alt="Cabinet">';
    } else if (endswith($resultRow["name_"], "Desk")) {
        $imageTag = '<img src="desk.png" alt="Desk">';
    } else if (endswith($resultRow["name_"], "Bedframe")) {
        $imageTag = '<img src="bedframe.png" alt="Bedframe">';
    } else if (endswith($resultRow["name_"], "Stool")) {
```

```
        $imageTag = '<img src="stool.png" alt="Stool">';
    } else if (endswith($resultRow["name_"], "Table")) {
        $imageTag = '<img src="table.png" alt="Table">';
    } else {
        $imageTag = "";
    }
    echo "<tr><td>" . $imageTag . "</td>";
    for ($colNum = 0; $colNum < $numCols; $colNum++) {
        echo "<td>" . $resultRow[$colNum] . "</td>";
    }
    echo "</tr>";
}
echo "</tbody></table>";

$result->closeCursor();
$result = null;
$conn   = null;
            ?>

</body>

</html>
```

query.php

```
<!-- -*-html-*- -->
<!DOCTYPE html>
<html lang="en-US">

<head>
    <title>Fantastic Furniture</title>
    <style>
        table {
            border-collapse: collapse;
        }

        table,
        th,
        td {
            border: 1px solid gray;
        }
    </style>
</head>

<body>
```

```php

    <?php
require 'conn.php';
$queries  = array(
    "SELECT M.designerID, COUNT(C.modelNumber) FROM Designer D LEFT OUTER JOIN make M ON ( D.designerID = M.designerID ) INNER JOIN contains_ C ON ( M.setID = C.setID ) GROUP BY M.designerID;",
    "SELECT AVG(leadTime) FROM Supplier S, DistributionCenter C, canOrderFrom O WHERE S.supplierID = O.supplierID AND C.centerID = O.centerID AND S.country = C.country;",
    "SELECT DISTINCT O.supplierID FROM Bedframe B, describes D, features_Feature F, canOrderFrom O, stocks S WHERE B.sku = D.sku AND D.modelNumber = F.modelNumber AND F.description = 'Claw Feet' AND O.centerID = S.centerID AND S.sku = B.sku AND O.leadTime < 7.0;",
    "SELECT COUNT(*) FROM stocks S, DistributionCenter D WHERE S.centerID = D.centerID AND D.name_ = 'Harold and His Big Ass Warehouse';",
    "SELECT DISTINCT C.setID FROM contains_ C WHERE (SELECT COUNT(DISTINCT I.sku) * C.count_ FROM Table_ I, describes D WHERE D.sku = I.sku AND D.modelNumber = C.modelNumber) >= 1 AND (SELECT COUNT(DISTINCT I.sku) * C.count_ FROM Chair I, describes D WHERE D.sku = I.sku AND D.modelNumber = C.modelNumber) > 2;"
);
$queryNum = intval($_GET["query"]) - 1;
assert($queryNum >= 0 && $queryNum <= 5, "Invalid query number!");
if ($queryNum < 5) {
    $query = $queries[$queryNum];
} else {
    $query = $_POST["userQuery"];
}
assert(preg_match("/\b(?:CREATE|ALTER|DROP|RENAME)\b/i", $query) === 0, "Query cannot contain CREATE, ALTER, DROP, or RENAME statements!");
$result  = $conn->query($query);
$numCols = $result->columnCount();

echo "<p><strong>Query:</strong> <code>" . $result->queryString . "</code></p>";
try {
    $output = "<table><thead><tr>";
    for ($colNum = 0; $colNum < $numCols; $colNum++) {
        $colMeta = $result->getColumnMeta($colNum);
        $output  = $output . "<th><code>" . $colMeta["name"] . "</code> [<code>" . $colMeta["native_type"] . "</code>]</th>";
    }
    $output = $output . "</tr></thead><tbody>";
    foreach ($result->fetchAll() as $resultRow) {
        $output = $output . "<tr>";
```

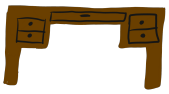
```
        for ($colNum = 0; $colNum < $numCols; $colNum++) {
            $output = $output . "<td>" . $resultRow[$colNum] . "</td>";
        }
        $output = $output . "</tr>";
    }
    $output = $output . "</tbody></table>";
    echo $output;
}
catch (PDOException $e) {
    echo "<p>Query completed.</p>";
}

$result->closeCursor();
$result = null;
$conn   = null;
            ?>

</body>

</html>
```

Resources

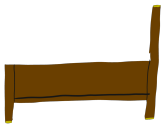
chair.png

cabinet.png

desk.png



bedframe.png



stool.png

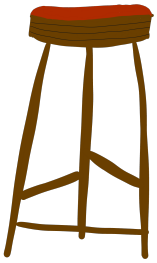


table.png