

Name: Jin Lin

Course: Data Mgt and Database Design

Course ID: INFO6210 18650 SEC 05

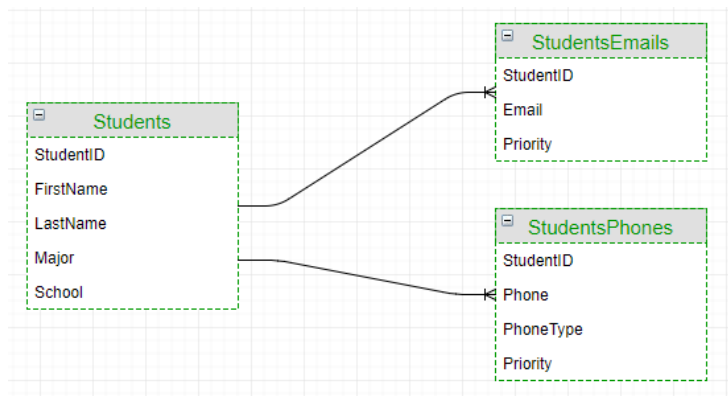
Instructor: Yusuf Ozbek

## Beginning Database Design Solutions

### Chapter 7

#### Exercise 1:

- a. In 1NF, each column should have a distinct name, a single datatype and a single value. This list does not obey the rules.



b.

#### Exercise 2:

- a. The items column contains multi-values.
- b. The following table is in 1NF, and the primary key is the combination LocationID/Item

Location ID	Location	Item
1	Grocery Store	milk
1	Grocery Store	Egg
1	Grocery Store	Bananas
2	Office Supply Store	Paper
2	Office Supply Store	Pencil

2	Office Supply Store	Diving rod
3	Post Office	Stamps
4	Computer Store	Flash drive
4	Computer Store	8 `` floppy disks

### Exercise 3:

- The location column depends on locationID column.
- It should be separated into two tables.

Location table

LocationID	Location
1	Grocery Store
2	Office Supply Store
3	Post Office
4	Computer Store

Items table

Location ID	Item
1	milk
1	Egg
1	Bananas
2	Paper
2	Pencil
2	Diving rod

3	Stamps
4	Flash drive
4	8 `` floppy disks

#### Exercise 4:

- a. The department column transitively depends on the project columns.
- b. It should be separated into two tables.

Employee-Project table

Employee	Project
Bill Michaels	Network Routing
Mandy Ponem	Network Routing
Mike Mix	Net Services Analysis
Deanna Fole	Survey Design
Julie Wish	Survey Design
Alice Most	Work Assignment
Josh Farfar	Work Assignment

Project-Department table

Project	Department
Network Routing	Network Lab
Net Services Analysis	Human Factors
Survey Design	Human Factors

Work Assignment	Network Lab
-----------------	-------------

### Exercise 5:

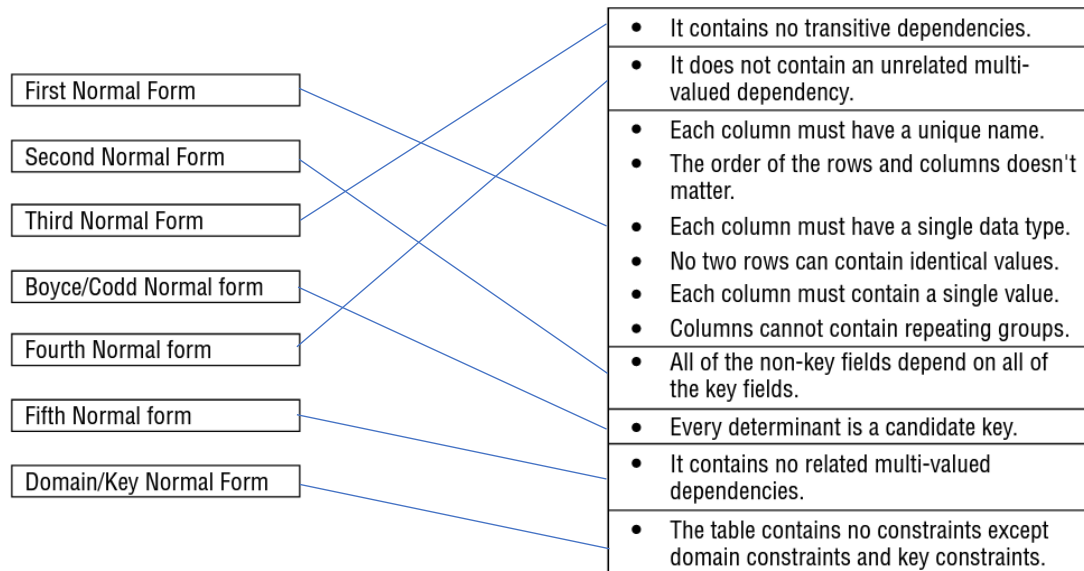
- The list contains no related multi-values dependencies.
- This list should be converted into three tables.

Person	Food
Alice	Muffins
Alice	Omelets
Alice	Pancakes
Bob	Muffins
Bob	Omelets
Bob	Pancakes
Cyndi	Omelets

Person	Tools
Alice	Muffin tin
Alice	Omelet pan
Alice	Pancake griddle
Bob	Omelet pan
Cyndi	Muffin tin
Cyndi	Pancake griddle

Tool	Food
Muffin	Muffin tin
Omelet	Omelet pan
Pancake	Pancake griddle

### Exercise 6:



## Beginning MySQL

### Chapter 4

#### Exercise 1:

Columns, rows and a primary key.

#### Exercise 2:

Each column can only have one data type and for each field, it can only have one value.

Therefore, each row should have the same number of columns. Besides, all rows should be different.

#### Exercise 3:

In a one-to-many relationship, a row on the one side can relate to many rows on the many side, while a row on the many side can only relate to one row on the one side. In a many to many relationship, a row on both many sides can relate to many rows on another many side.

**Exercise 4:**

Normalize the data according to the rules.

**Exercise 5:**

A foreign key table must be implemented to a many-to-many relationship, by creating one-to-many relationships to both many sides.

**Beginning SQL****Chapter 4****Exercise 1:**

ISBN is unique to identify a book and it could be made to be a primary key. And the Structure should be divided into 6 tables. Books to publishers is one to one relationship. Books to authors and sellers is one to many relationship.

Books table:

ISBN	Varchar
BookTitle	Varchar
YearPublished	Int
PublisherCode	Int

Publisher table:

PublisherID	Int
PublisherName	Varchar
PublisherAddress	Varchar

Book-Authors table:

ISBN	Varchar
AuthorID	int

Author table:

AuthorID	Int
AuthorName	Varchar
AuthorAddress	Varchar

Book-Seller table:

ISBN	Varchar
BookSellerID	int

Seller table:

SellerID	Int
SellerName	Varchar
SellerTelNo	Varchar

**Exercise 2:**

```
Alter table favcategory
add constraint favcategory_cat_fk
foreign key (categoryID)
references category(categoryID);




Alter table favcategory
add constraint favcategory_member_fk
foreign key (MemberID)
references memberdetails(MemberID);
```

## Chapter 6

### Exercise 1:

```
1 • SELECT
2     c.category,
3     count(FilmId) 'Number of films',
4     DVDPrice * 1.1 'Cost of DVD'
5 FROM moviedb.films f
6 inner join category c on c.CategoryId = f.CategoryId
7 where f.AvailableOnDVD = 'Y'
8 group by c.Category
9 having count(Filmid) = 1;
```

<



Result Grid  Filter Rows:  Export:  Wrap Cell Content: 

	category	Number of films	Cost of DVD
	Historical	1	17.60
	Horror	1	11.00
	Romance	1	14.30
	Sci-fi	1	14.30
	Thriller	1	3.30
	War	1	14.30

### Exercise 2:

```
1 • SELECT
2     c.Category,
3     IFNULL(MAX(f.rating), '') 'Highest rating',
4     IFNULL(MIN(f.Rating), '') 'Lowest rating'
5 FROM
6     moviedb.category c
7     LEFT JOIN
8     films f ON f.CategoryId = c.CategoryId
9 GROUP BY c.Category
10 ORDER BY c.CategoryId ASC;
11
```

<

Result Grid  Filter Rows:  Export:  Wrap Cell Conte

	Category	Highest rating	Lowest rating
	Thriller	1	1
	Romance	4	4
	Horror	2	1
	War	5	2
	Sci-fi	5	1
	Historical	5	3
	Comedy		
	Film Noir		