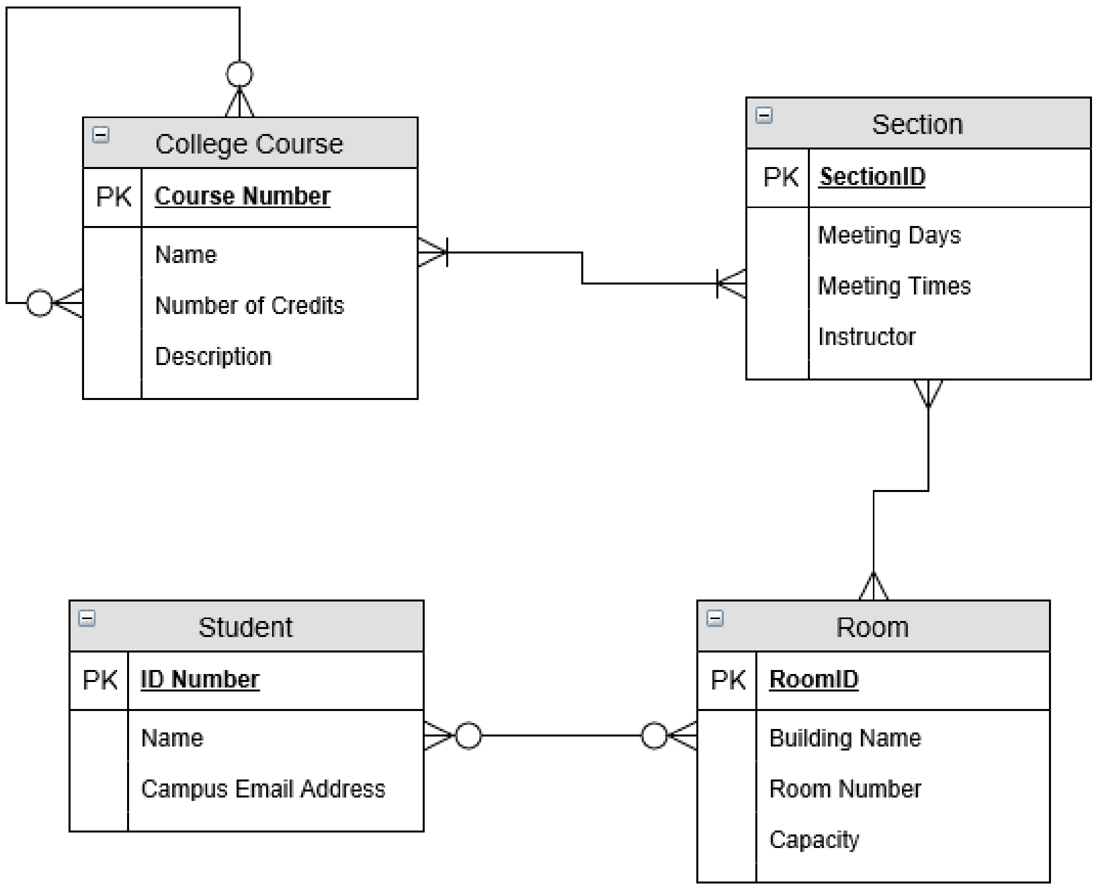
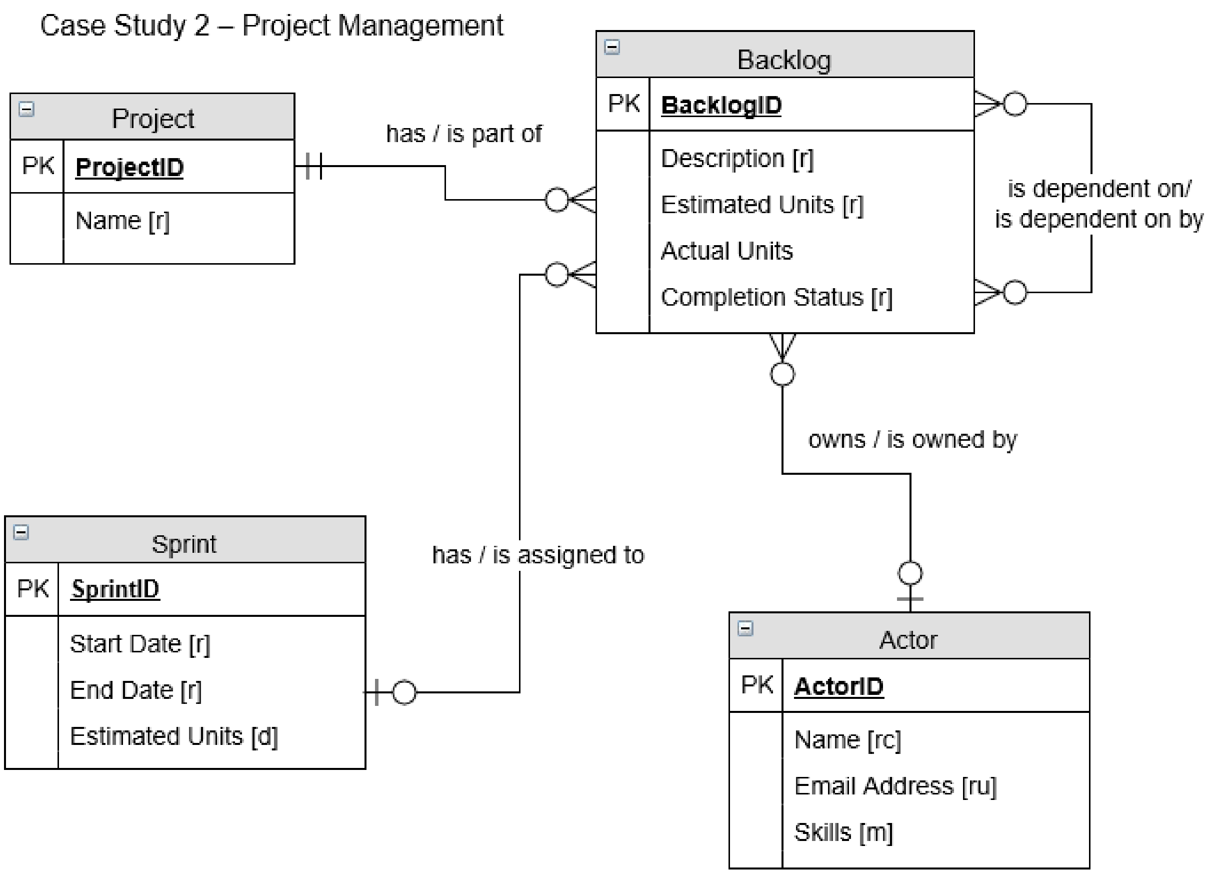
Beverlyn Tucker

IST659_M406 Week2_Lab2

Lab 02 – Conceptual Modeling

Case Study 1 – Obligatory College Classes Modeling





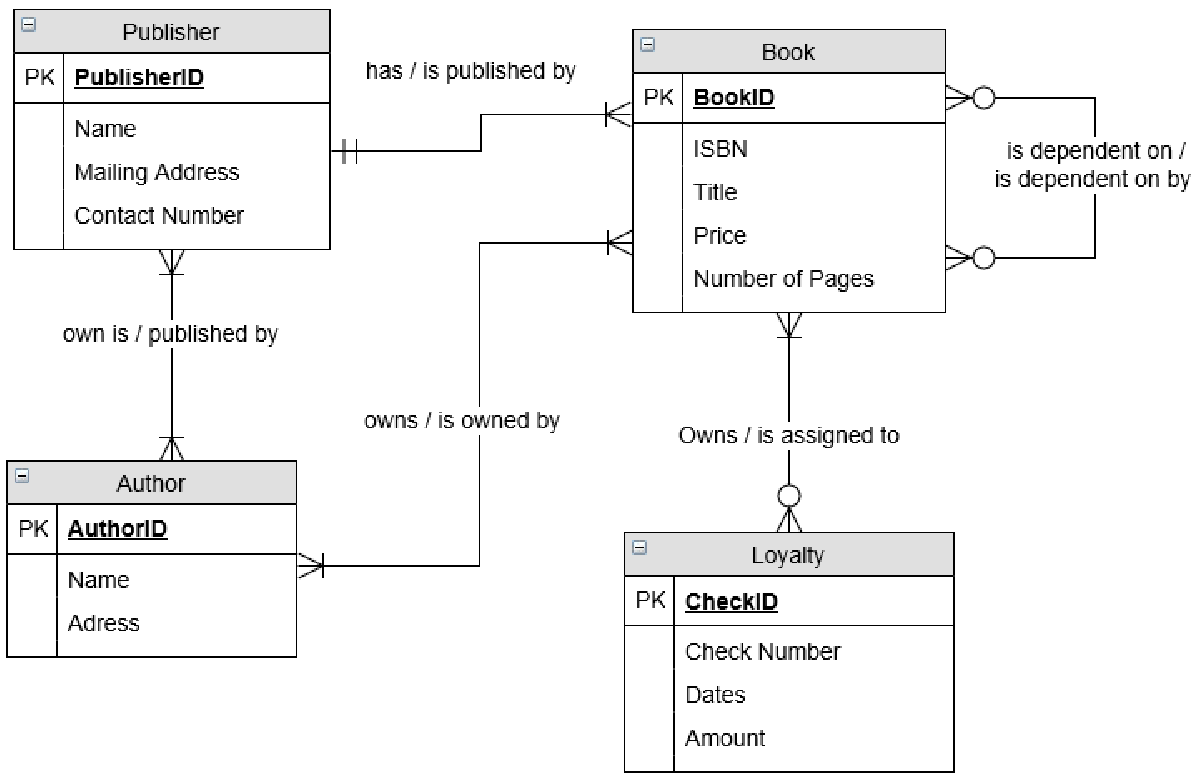
Case Study 3 – Book Publishing Database

In this example, you'll identify the entities, attributes, and relationships yourself and model them using draw.io. Each publisher has a unique name; a mailing address and telephone number are also kept on each publisher. A publisher publishes one or more books; a book is published by exactly one publisher. A book is identified by its ISBN, and other attributes are title, price, and number of pages. Each book is written by one or more authors; an author writes one or more books, potentially for different publishers. Each author is uniquely described by an author ID, and we know each author's name and address. Each author is paid a certain royalty rate on each book he or she authors, which potentially varies for each book and for each author. An author receives a separate royalty check for each book he or she writes. Each check is identified by its check number, and we also keep track of the date and amount of each check.

Entity	**Attributes**
Publisher	Name
	Mailing Address
	Contact Number
Book	ISBN

	Title
	Price
	Total Pages
Author	Name
	Address
Loyalty	Check Number
	Dates
	Amount

Case Study 3 – Book Publishing Database

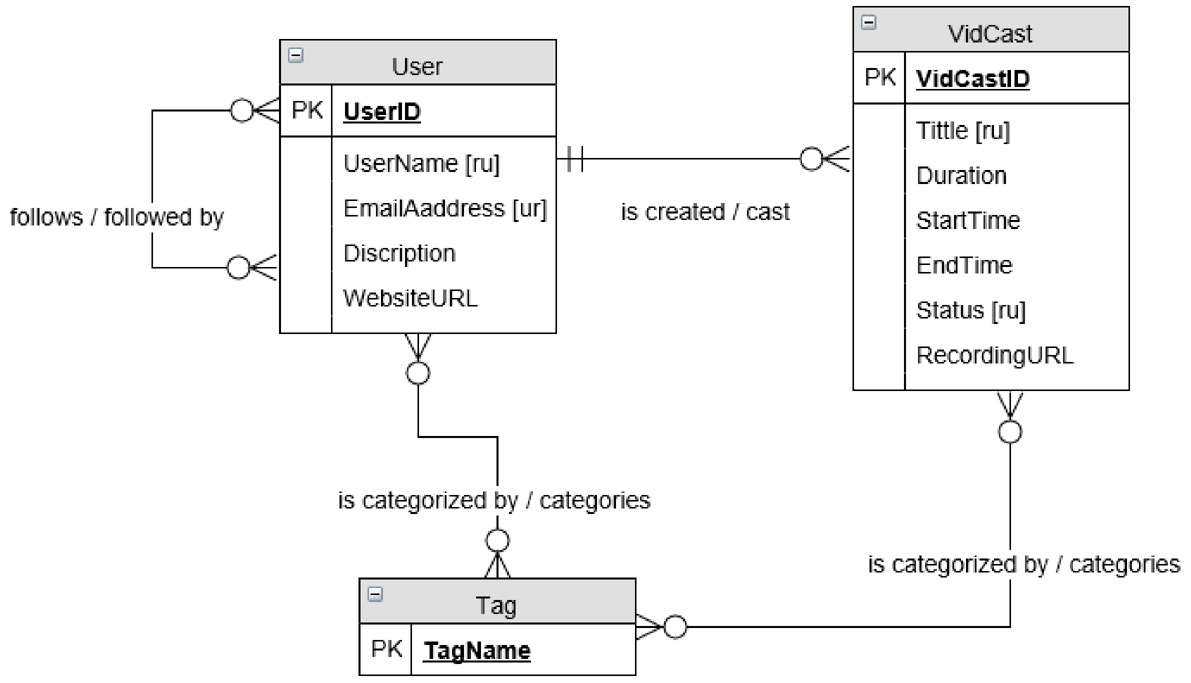


Part 2 – VidCast Conceptual Model

(think unary relationship!). To help users find one another, each user can add categorizing tags to their profile. Each of these tags can be applied to many other users as well. Vidcasts are broadcasts of a video stream, identified by a system-generated VidCast ID. A vidcast must have a title. A vidcast may have a start date and time and a projected duration. This duration is replaced with the actual duration. A Vidcast can also be recorded. If it is, the recording will be stored on a secure web service such as Amazon Web Services S3. We will need to log the URL of this recoding with the vidcast. Each vidcast is made by exactly one user, but each user can optionally create many vidcasts. The user can tag each vidcast using the same tag list as is used for user tagging. Because vidcasts can be scheduled ahead of time, each vidcast requires a status (Scheduled, Started, Stopped, Cancelled are examples of these statuses).

Entities	**Attributes**
VIDCAST	Tittle [ru]
	Duration
	StartTime
	EndTime
	Status [ru]
	Recording/URL
User	UserName
	EmailAddress
	Discription
	Web site/URL
Tags	TagName

Part 2 – VidCast Conceptual Model


User
PK UserID
UserName [ru]
EmailAaddress [ur]
Discription
WebsiteURL
follows / followed by
is created / cast
VidCast
PK VidCastID
Tittle [ru]
Duration
StartTime
EndTime
Status [ru]
RecordingURL
is categorized by / categories
is categorized by / categories
Tag
PK TagName