LightHouse Academy

5.4 DB設計

- データベースの設計
- 概念設計
- 論理設計
- 物理設計

www.lighthouseit.academy

目次

1 データベースの設計

2 概念設計

3 論理設計

4 物理設計

LightHouse Academy

データベースの設計

- データベースの設計とは、データベースでデータを管理できるように、現実の世界を抽象化してデータモデルを作成することです。

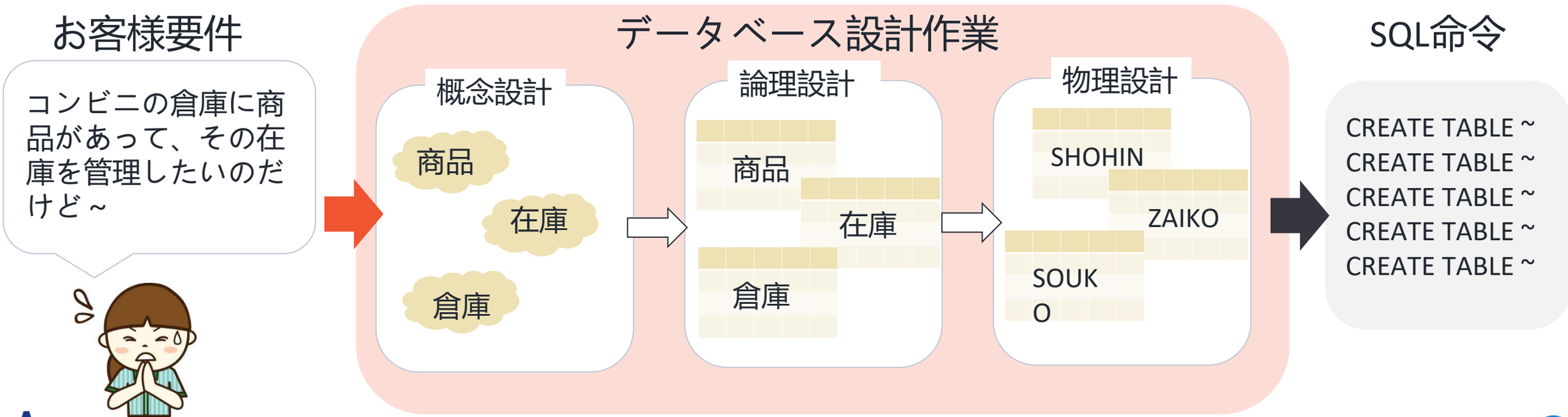
- 簡単にいうと、作りたいものを考えたときに、どのテーブルを用意して、各テーブルの列には、何を入れるのか、そして各列の属性（型、デフォルト値）はどうするのかを考えることをしていくと思ってください。

データベース設計の目的

- 目的は、一言でいうと、「無駄のないテーブル」を作成するためです。
- 「複数個所に同じ情報がある」、「削除するとデータの整合性が保たれないのに削除がOK」などの潜在的な問題を洗い出して、これらが発生しないようなテーブルを設計・作成します。
- データベースを設計することは、業務効率化と業績の向上につながります。 必要な情報をすぐに利用できるデータベースを設計することで、社員の無駄な作業を削減し、売上につながる情報をリアルタイムに活用できるようになります。

データベース設計の流れ

- データベース設計は、大きく分けると3つの段階を分かれます。
  - 概念設計
  - 論理設計
  - 物理設計
- それぞれの工程は、次のページで解説していきますが、まずは、概要をイメージしておきましょう。



LightHouse Academy

目次

1 データベースの設計

2 概念設計

3 論理設計

4 物理設計

概念設計

- 概念設計は、管理すべき情報はどうのようなものかを整理します。
  データベースやシステムに関することは考えず、
  要件に登場する情報だけをざっくりと把握します。

- 例えば、在庫データベースであれば、「商品名」、「在庫数」などがあることを明確にします。
  また、情報同士に関連がある時は、どのような関係かも合わせて整理します。

- 概念設計の流れ

概念設計ですること

- 概念設計では、要件を実現するために、抽象的な概念としてどのような「情報の塊」を管理しなければなりませんでした。

- 情報の塊のことをエンティティ(Entity)といい、
  エンティティは、通常、複数の属性(attribute)を持っている。

- さらに、エンティティ同士にどのような関連があるかも、この
概念設計で明らかにします。

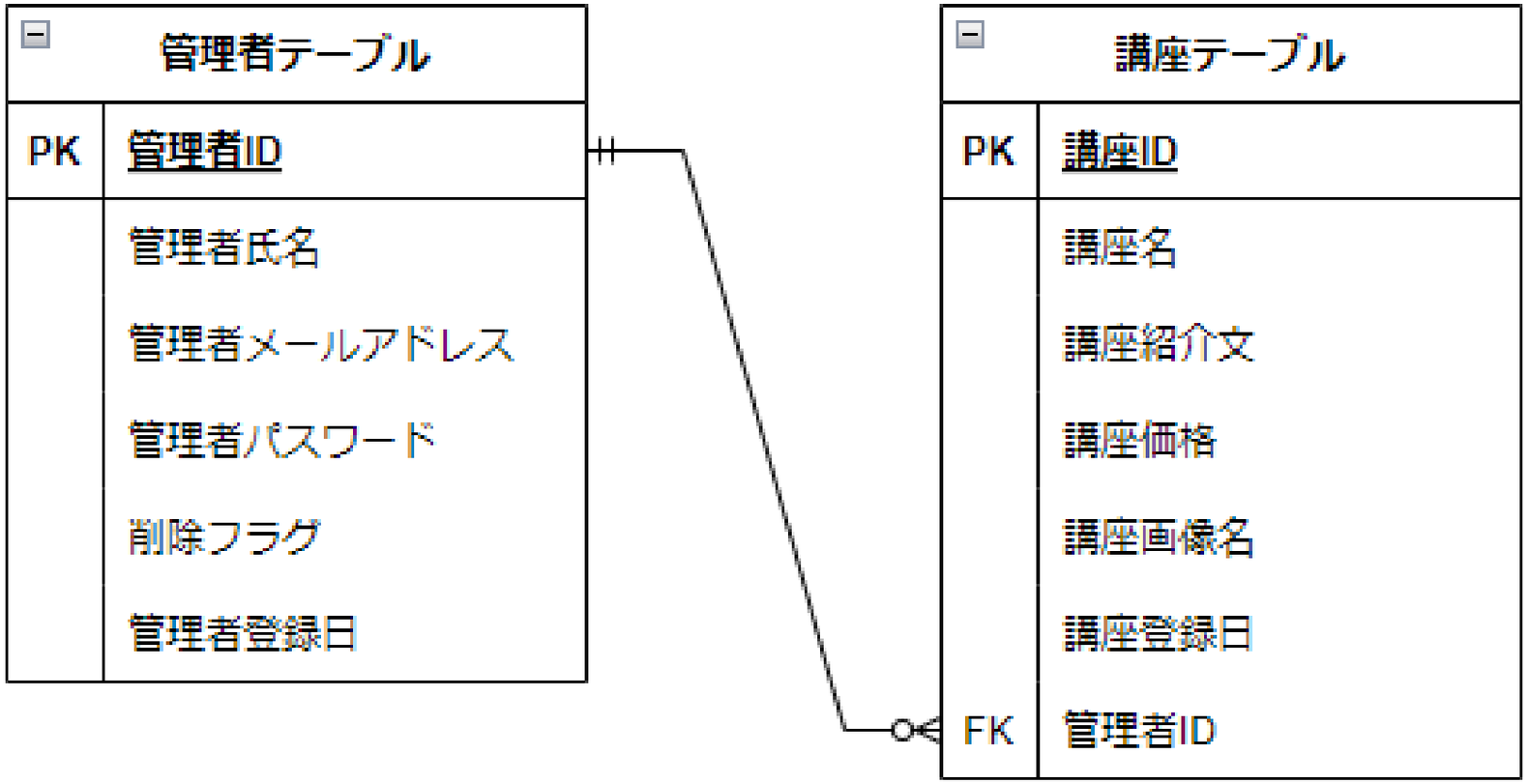
概念的なもののイメージをつかむヒント

- エンティティ：「テーブル」のようなもの
- 属性　　：テーブルの「列」のようなもの
- 関連　　：「リレーションシップ」のようなもの
- 例えば、書店の在庫管理を概念モデルで表す場合
  - エンティティ：「書籍情報」、「在庫情報」
  - 属性　　：書籍Entity⇒タイトル、価格
  - 関連　　：在庫情報に、どの書籍が何冊あるかという情報があるので「在庫情報」と「書籍情報」は関連がある。

ER図

- 概念設計の成果は、ER図 (ERD:entity-reratoinship-diagram)と呼ばれる図にまとめる。

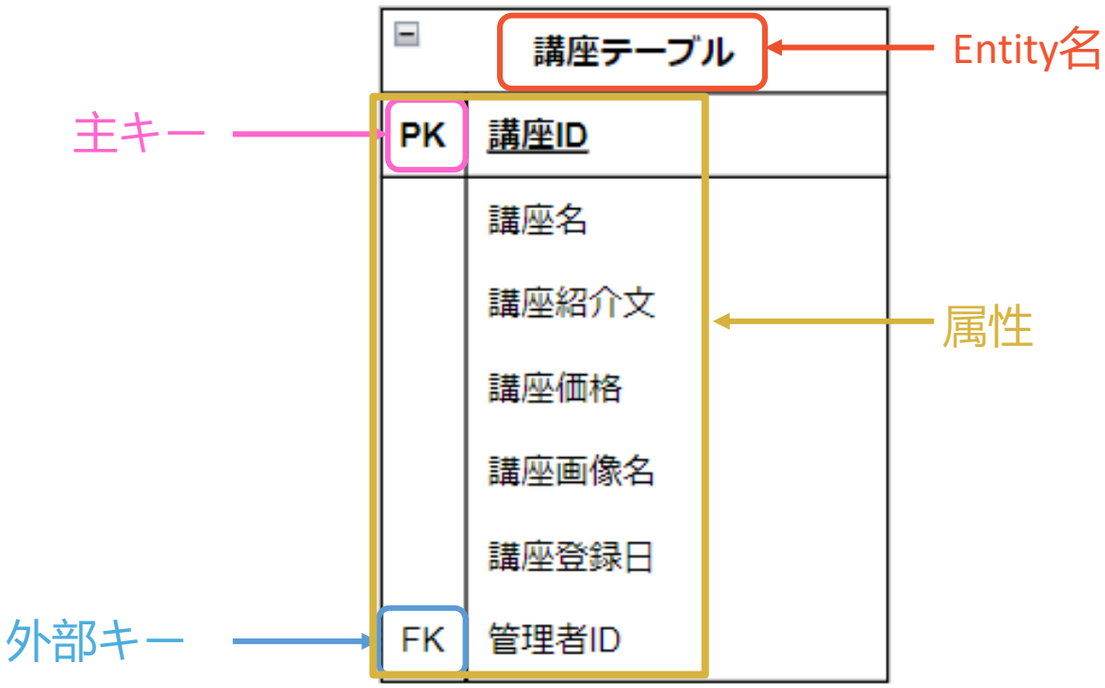
- ER図にまとめることで、「エンティティ」、「属性」、「リレーションシップ」を客観的にみることができる。



ER図の記述ルール

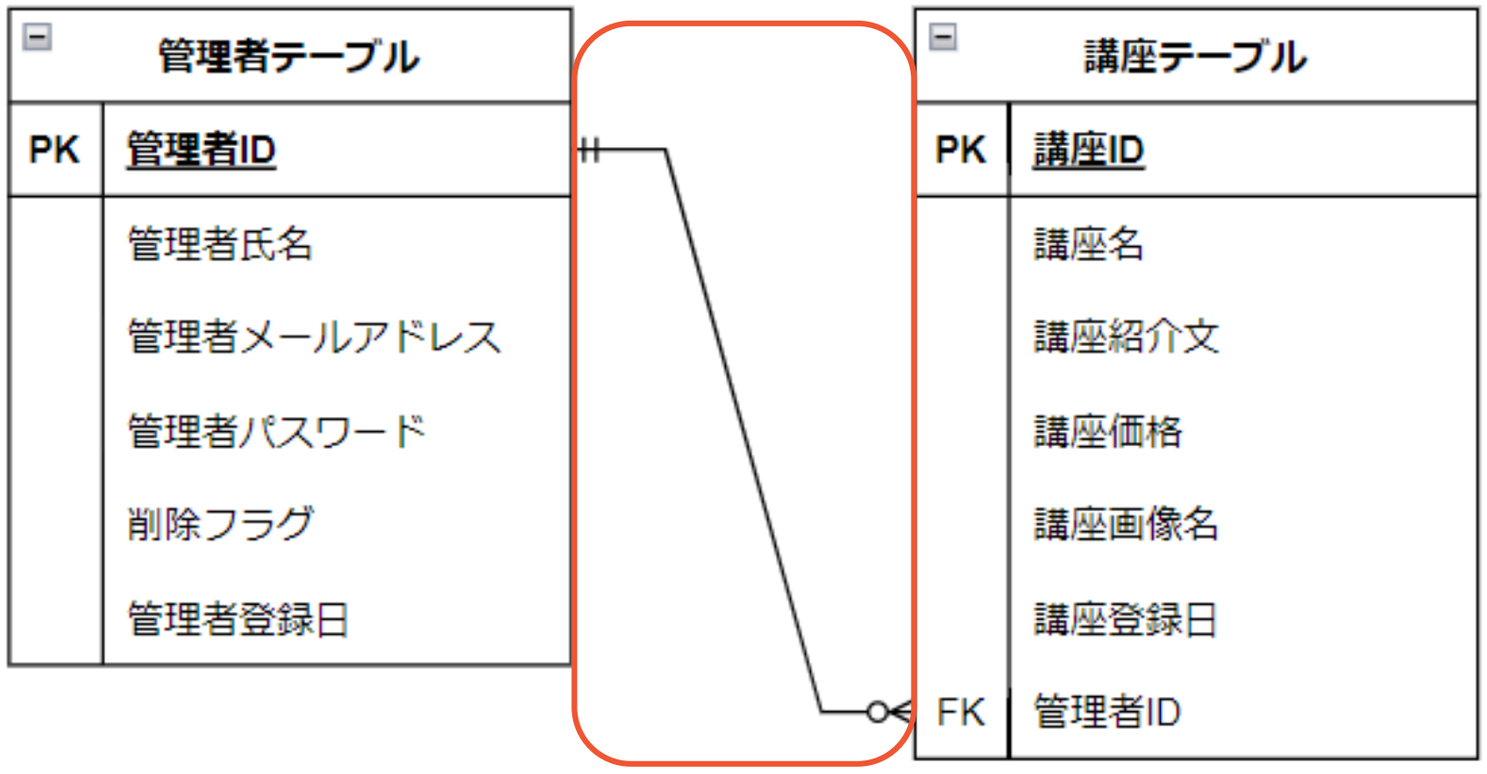
- ER図に登場する四角形は「エンティティ」を表している。
  エンティティ内に記載するもは、以下である。

  - Entity名
  - 主キー
  - 属性
  - 外部キー（あれば）

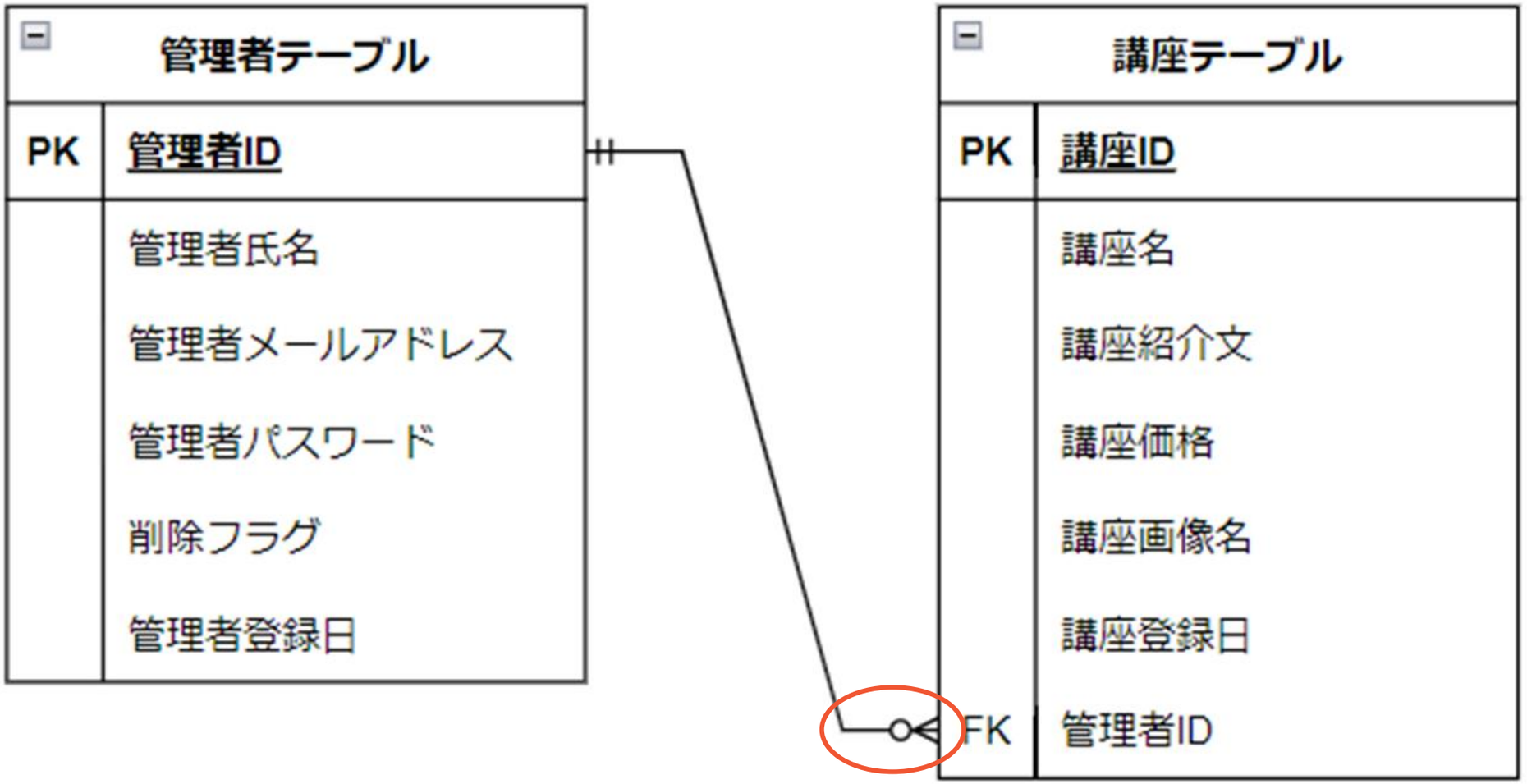


次へ

- リレーション
  エンティティ間にリレーションシップがある場合には、
  エンティティ同士を線でつなぎます。
  下記の図は、管理者テーブルと講座テーブルがリレーション
  関係にあることを表現しています



- カーディナリティ（多重度）
  カーディナリティとは、「1対1」、「1対N」、「N対N」など、リレーションの詳細（数量的な関係）を表現する記号のことです。



次へ

前へ

- カーディナリティの書き方
  カーディナリティには、「IE記法」と「IDEF1X」記法がありますが、今回は、「IE記法」についてご紹介します。

IE記法は、「〇」、「|」、「鳥の足（三つ股の線」という3つの記号を組み合わせて表現していきます。

記号	意味
〇	0
|	1
	N

次へ

- カーディナリティの記号を組み合わせることで様々な表現をすることができます。いくつか例を見てみましょう。

1対1

1又は0対N（0以上）

次へ

前へ

1対N以上

N（0以上）対N（0以上）

目次

1 データベースの設計

2 概念設計

3 論理設計

4 物理設計

論理設計

- 先ほど、勉強したER図は、あくまでも概念の世界における理想的なエンティティ構造を表現しているに過ぎないため、このままの姿でデータベースに格納できるとは限りません。

- そこで、利用する予定のデータベースが扱いやすい構造にエンティティを変形する作業を行います。

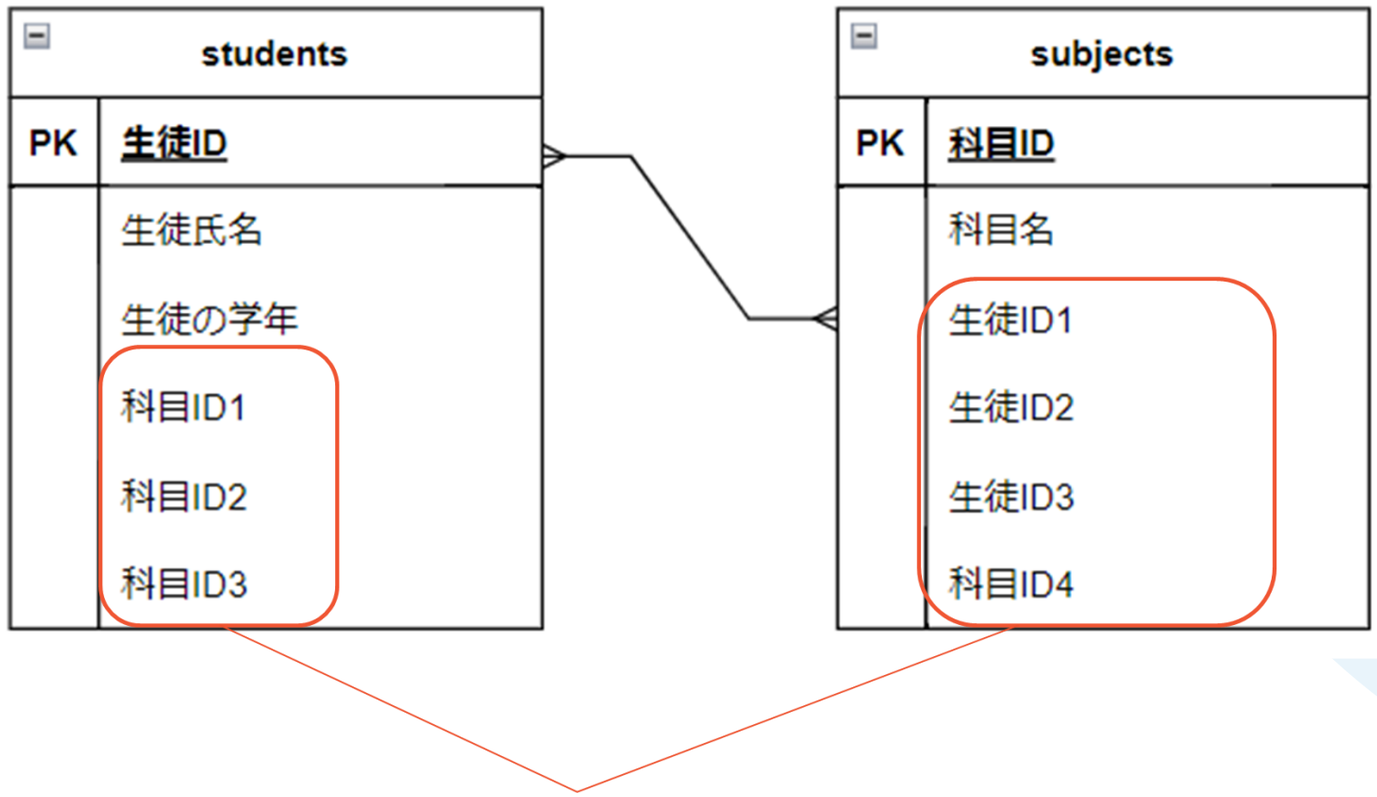
- 論理設計の流れ

多対多の分解

- 多対多のリレーショナルでは、テーブルの間に、中間テーブルを作成します。
- テーブル間に中間テーブルを作成しない場合は、設計時に複数のカラムが必要となり、カラムが肥大化していきます。
- 例えば、「一人の学生は複数の教材を履修する」、「1つの科目は複数の学生が履修する」という要件で学生テーブルと科目テーブルを作成した場合、この2つのテーブルは、多対多の関係になります。次のページで見てみましょう。

次へ

前へ



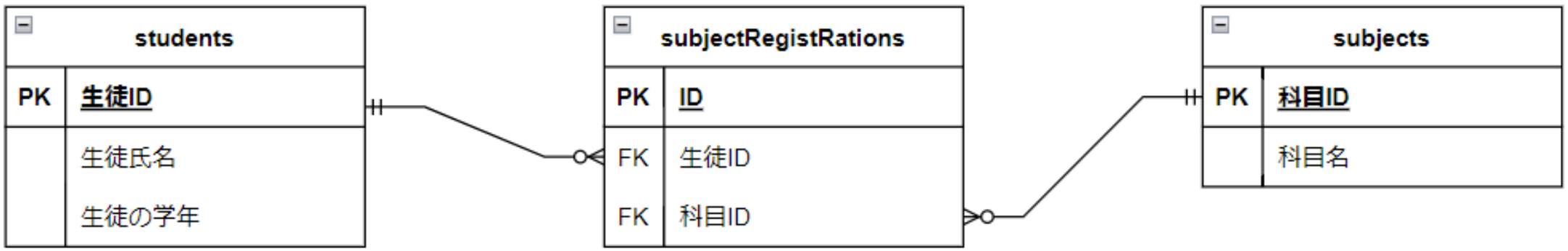
肥大化する

- 学生のテーブルに各科目のカラムを作成した場合を考えてみます。
- すると、科目テーブルに新たに科目が増える度に、学生テーブルに追加された科目のカラムを追加しなければならないため、好ましい設計とは言えません。ここで使用されるのが、中間テーブルです。

次へ

前へ

- 中間テーブルには、学生・科目テーブルのIDを格納するカラムだけを用意します。
  これにより、1人の学生に対して中間テーブルが1対多の関係となります。



- そのため、生徒が科目を登録する場合、中間テーブルには生徒のIDと科目IDのみを挿入するだけで生徒は複数の科目が登録でき、科目側からも1つの科目に対して中間テーブルを経由することで履修している複数の学生を参照することが可能になります。

次へ

- イメージ図

students

生徒ID	生徒氏名	生徒の学年
1	東京太郎	3
2	東京花子	4
3	埼玉次郎	1

subjectRegistRations

ID	生徒ID	科目ID
1	1	2
2	1	1
3	2	3

subjects

科目ID	科目名
1	化学
2	数学
3	国語

キーの整理

- ここでは、出そろったすべてのエンティティの主キーについて整理と確認をします。
- 主キーを持たないエンティティには、管理をしやすくするために、人工的な主キー（人工キー）を追加します。
- その他、不適切な主キーを持つエンティティがないかを確認します。
- 主キーが備える3つの特性
  - 非NULL性：必ず何らかの値を持っている
  - 一意性　　：他と重複しない
  - 不変性　　：一度決定されたら値が変更されることがない
    （主キーは、一貫して同じ1行を指す）

正規化

- 論理設計における最も中心的な作業は、正規化[normalizatoin]の作業です。
- 正規化とは、「矛盾したデータを格納できないように、テーブルを複数に分割していく作業」のことをいいます。
- この正規化の考え方は、非常に難しいです。皆さんは、正規化とは何かということを説明できれば良いでしょう。

目次

1 データベースの設計

2 概念設計

3 論理設計

4 物理設計

LightHouse Academy

物理設計の流れ

- 論理設計後、どのDBMS製品を利用するかを確定した上で、行うのが、「物理設計」です。

- DBMS製品がサポートしている型や制約、利用するハードウェア等の制約を考慮し、全テーブルについて詳細な設計を確定させます。

テーブル定義書

- 物理設計の成果は、テーブル定義書と呼ばれる書類にまとめます。

テーブル項目定義

No.	項目名(論理名)	項目名(物理名)	キー	論理キー	データ型	サイズ	デフォルト値	NULL 制約	備考
1	講座 ID	lesson_id	PK		bigint			NOT NULL	
2	講座開始日	start_date			date				
3	講座開始時間	start_time			time				
4	講座終了時間	finish_time			time				
5	講座名	lesson_name			varchar	255			
6	講座詳細	lesson_detail			varchar	255	current date		
7	講座金額ド	lesson_fee			int				
8	講座画像名	image_name			varchar	255			
9	講座登録日	register_date			timestamp				
10									
11									
12					-				
13					-				
14					-				
15					-				
16					-				
17					-				
18					-				
19					-				

次へ

テーブルインデックス定義

No.	Index 種類	Index 名	構成要素
1	primary key	lesson_pkey	lesson_id
2			
3			
4			
5			
6			
7			

テーブル関連定義

No.	外部キー名	カラムリスト	参照先テーブル名	参照先カラムリスト
1	admin	admin_id	admin	admin_id
2				
3				
4				
5				
6				
7				
8				
9				
10				
11				
12				
13				
14				
15				
16				

次へ

- 論理名：「プログラムにおける人間向けの名称」
- 物理名：「プログラムにおけるコンピューター向けの名称」
- **「物理名」**も**「論理名」**もプログラム言語やアプリケーション開発などで使われる用語ですが、**「物理名」**はデータベースがテーブルを識別するための**「本当の名前」**を意味しています。

それに対して、**「論理名」**というのは**「人間にとって識別しやすく分かりやすい名前」**や**「本当の名前に対するあだ名(呼びやすい名前)」**を意味しているという違いがあります。

次へ

- 倫理名と物理名の例となります。

No.	項目名(論理名) プログラムにおける人間向けの名称	項目名(物理名) プログラムにおけるコンピューター向けの名称
1	講座 ID	lesson_id
2	講座開始日	start_date
3	講座開始時間	start_time
4	講座終了時間	finish_time
5	講座名	lesson_name
6	講座詳細	lesson_detail
7	講座金額ド	lesson_fee
8	講座画像名	image_name
9	講座登録日	register_date

まとめ

Sum Up

1.データベース設計の流れ：
概念設計⇒論理設計⇒物理設計

2.概念設計：
　①ER図の書き方

3.論理設計：
　①中間テーブル
　②キー整理
　③正規化

4.物理設計：テーブル定義書

Light in Your Career.

THANK YOU!