データベースS

第8回　リレーションスキーマの設計(2)

システム創成情報工学科　尾下 真樹

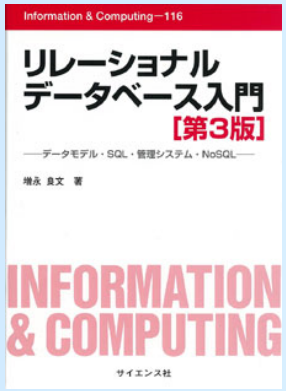
2018年度 Q2

今日の内容

- 前回の復習
  - 正規化の必要性
  - 関数従属性と多値従属性
- 正規形と正規化
  - 第２正規形
  - 第３正規形
  - ボイス･コッド正規形
  - 第４正規形
  - 第５正規形

教科書

- 「リレーショナルデータベース入門［第3版］」
  増永良文 著、サイエンス社
  - 4章（4.8～4.13節）
- 「データベースシステム」
  北川 博之 著、昭晃堂 出版
  - 4章 55～82ページ



  - 教科書により、正規形の定義の表記が異なるが、本講義では、両方の内容を説明

前回の復習

前回の復習

- 正規化の必要性
- 関数従属性と多値従属性

正規形と正規化

- 正規形
  - データの更新・修正・削除などを行っても不整合が生じないようなリレーションスキーマ
  - 正規形を満たさないスキーマ
    - 1つのリレーションに複数の事象の情報が格納されている → 不整合が生じる原因となる
- 正規化
  - 正規形を満たさない原因となっている関数従属性・多値従属性に注目して、スキーマを分解

なぜ、正規化が必要か?

- 更新不整合(更新時異常)が生じる
  - 正規化されていないリレーションは、データを更新(挿入、修正、削除)しようとした時に問題(データの不整合)が生じる場合がある
- 正規化を行うことで解決
  - 更新時にデータの整合性が自動的に保たれる(整合性を壊すようなデータの追加・修正ができないような)データベースができる

更新不整合

- 更新不整合の種類
  - 修正不整合(修正時異常)
  - 挿入不整合(挿入時異常)
  - 削除不整合(削除時異常)

正規化

- 正規形を満たさないリレーションスキーマがあるときに、正規形を満たすように分解する

営業

商品番号	顧客番号	社員番号	販売価格
i1	c1	s1	100
i2	c1	s1	200

分解

販売

商品番号	顧客番号	販売価格
i1	c1	100
i2	c1	200

営業担当

顧客番号	社員番号
c1	s1
c2	s2

関数従属性と多値従属性

- 関数従属性　X → Y
  - 属性（の組）X が決まれば、属性（の組）Y が一意に決まる
- 多値従属性　X →→ Y
  - ある属性（の組） Xについて、いくつかの属性（の組）Yが存在すれば、必ず全ての XY（RS－XY）の組み合わせが存在する
    - RSはリレーションの全ての属性
  - 関数従属性は多値従属性の特殊なものと言える
    - Yが常に1種類のみ存在するもの

関数従属性の例

- 各顧客につき、担当する社員は一人だけとする

営業

商品番号	顧客番号	社員番号	販売価格
i1	c1	s1	
i1	c2	s2	
i2	c1	s1	
i2	c2	s2	210
i3	c1	s1	
i3	c2	s2	
i4	c1	s1	

例えば、顧客番号が c1 であれば、担当する社員番号は常に s1 になる

このスキーマには、

顧客番号 → 社員番号

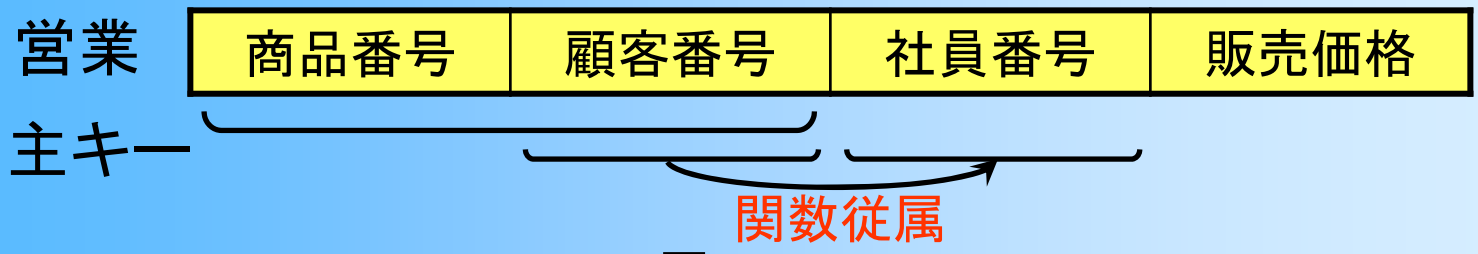
という関数従属性がある

主キー

関数従属

（関数従属の属性で分解できる）

関数従属性にもとづく分解の例



分解

販売

商品番号	顧客番号	販売価格
i1	c1	100
i1	c2	120
i2	c1	200
i2	c2	210
i3	c1	250
i3	c2	250
i4	c1	150

営業担当

顧客番号	社員番号
c1	s1
c2	s2

多値従属性の例

- プロジェクトごとに社員とミーティング日が決定
  - 例: p1 は、社員{e1,e2}、ミーティング日{月,木}

プロジェクト

プロジェクト番号	社員番号	ミーティング日	
p1	e1	月曜日	t
p1	e2	月曜日	w
p1	e1	木曜日	v
p1	e2	木曜日	u
	e1	月曜日	
	e3	月曜日	

t(x, y_1, z_1), w(x, y_2, z_2) というタプルが存在すれば 、必ず v(x, y_1, z_2), u(x, y_2, z_1) のようなタプルも存在する

X, Y は複数の属性の組でも良く、Z＝RS－XY であることに注意

多値従属性にもとづく分解の例

プロジェクト

プロジェクト番号	社員番号	ミーティング日

主キー

プロジェクト番号 →→ 社員番号 | ミーティング日

分解

参加社員

プロジェクト番号	社員番号
p1	e1
p1	e2
p2	e1
p2	e3

ミーティング日

プロジェクト番号	ミーティング日
p1	月曜日
p1	木曜日
p2	月曜日
p2	金曜日

関数従属性の性質

- その属性にどのような値が入るかという条件によって決まる
  - スキーマだけからは分からず、データベースに格納されるデータについての予備知識が必要
    - 今回の例では、1件の顧客につき担当者1人というきまりなど
  - スキーマ設計を行う時には、どのような関数従属性が存在するかを調べて、書き出す必要がある

正規形と正規化

正規形(復習)

- 更新時に不整合が発生しないような、整合性を保つリレーションスキーマの条件を定義
- 正規形の種類
  - 第１正規形
  - 第２正規形
  - 第３正規形
  - ボイス・コッド正規形
  - 第４正規形
  - 第５正規形

第１正規形→第５正規形まで、徐々に条件が厳しくなっていく

各正規形は、それよりも上の全ての正規形の条件を満たす

正規形と正規化

- 関数従属性や多値従属性にもとづいて、リレーションが正規形を満たすかどうかを判定
- 正規形を満たさないリレーションがあれば、正規形を満たすように、リレーションを分解（＝正規化）
- 段階的に正規化を行っていき、最終的には第５正規形まで満たすようにする

正規形と正規化

- 第２正規形
- 第３正規形
- ボイス･コッド正規形
- 第４正規形
- 第５正規形

第2正規形

- 定義（教科書「リレーショナルデータベース入門」）
  - 第１正規形を満たし、
  - 全ての非キー属性が候補キーに完全関数従属
    - 候補キーの一部の属性にのみ関数従属しない
- 別の定義（教科書「データベースシステム」）

  リレーションスキーマ RS において関数属性$X \rightarrow A$が成り立つとき、以下のどちらかの条件を満たす

  1. X は RS のどの候補キーの部分集合でもない
  2. A が素属性である

  - 素属性・・・候補キーの構成属性（非キー属性）

第２正規形を満たさない例

- 第２正規形を満たさない例
  - 各顧客を担当する従業員が決まっているとする

営業

商品番号	顧客番号	社員番号	販売価格
i1	c1	s1	100
i1	c2	s2	120
i2	c1	s1	200
i2	c2	s2	210
i3	c1	s1	250
i3	c2	s2	250
i4	c1	s1	150

主キー

関数従属

顧客番号→社員番号

第２正規形を満たさない例

- 第２正規形を満たさない例
  - 顧客番号が候補キーの部分集合になっている

営業

商品番号	顧客番号	社員番号	販売価格
i1	c1	s1	100
i1	c2	s2	120
i2	c1	s1	200
i2	c2	s2	210
i3	c1	s1	250
i3	c2	s2	250
i4	c1	s1	150

主キー

関数従属

キー以外に関数従属

分解のルール

- 関数従属性・多値従属性に注目した情報無損失分解
  - リレーション RS 中に、X → Y または X →→ Y の関数従属性・多値従属性があるとき、(X, Y) と (RS － Y) の2つのリレーションに分解できる
    - (X, Y) の主キーは X
    - (RS － Y) の主キーは元の RS と同じもの、となる
      - ただし Y が元の RS の主キーの属性の組の一部であれば、主キーの属性の組の Y を X で置き換える

第２正規形を満たすための分解

営業

商品番号	顧客番号	社員番号	販売価格

顧客番号→社員番号 の
関数従属性に基づき分解

販売

商品番号	顧客番号	販売価格

もとのリレーションから 社員番号 を取り除く
（顧客番号 は残す）

顧客担当

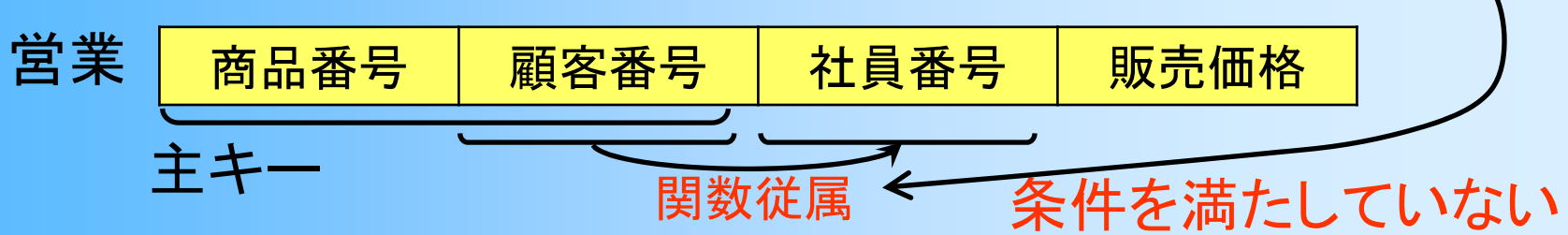
顧客番号	社員番号

顧客番号 と 社員番号 で
1つのリレーションとする

※ 分解後のリレーション名は適当に決める

正規形の判定と分解の注意

- 正規形の定義を満たす場合は問題ない
- 定義を満たさない従属性があれば、その従属性に注目して、リレーションを分解
  - 第１正規形を満たし、
  - 全ての非キー属性が候補キーに完全関数従属
    - 候補キーの一部の属性にのみ関数従属しない



正規形と正規化

- 第２正規形
- 第３正規形
- ボイス･コッド正規形
- 第４正規形
- 第５正規形

第3正規形の定義

- 定義（教科書「リレーショナルデータベース入門」）
  - 第2正規形を満たし、
  - 非キー属性が、候補キー以外の属性に関数従属しない
- 別の定義（教科書「データベースシステム」）

  リレーションスキーマ RS において関数属性 $X \rightarrow A$ が成り立つとき、以下のどちらかの条件を満たす

  1. X は RS の超キーである
  2. A が素属性である

第３正規形を満たさない例

- 第３正規形を満たさないリレーションの例
  - 各社員は1つの営業所に所属する、とする
    - 社員番号によって、営業所番号は一通りに決まる
  - このリレーションは第２正規形を満たす
    - 社員番号, 営業所番号は、商品番号に関数従属

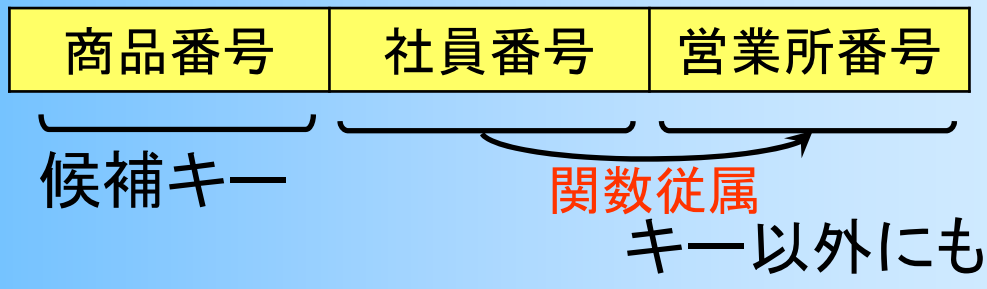
商品担当

商品番号	社員番号	営業所番号
i1	s1	o1
i2	s2	o2
i3	s2	o2
i4	s1	o1

候補キー

第３正規形を満たさない例

- このリレーションスキーマの問題点
  - 営業所番号が社員番号によって決まる
    - 社員番号→営業所番号 という関数従属性が存在
  - 挿入不整合、修正不整合、削除不整合がそれぞれ起こる
    - 社員の所属営業所の情報と商品担当の情報がひとつのリレーションに混在していることが原因



第３正規形を満たす分解

- 第３正規形を満たさないリレーションの例
  - 各社員は1つの営業所に所属する
  - 非キー属性である営業所番号が、候補キー以外の属性である社員番号に関数従属

社員番号→営業所番号

商品担当

商品番号	社員番号	営業所番号
i1	s1	o1
i2	s2	o2
i3	s2	o2
i4	s1	o1

候補キー

関数従属

第３正規形を満たすための分解

商品担当

商品番号	社員番号	営業所番号

候補キー

関数従属

社員番号→営業所番号 の
関数従属性に基づき分解

商品担当

商品番号	社員番号
i1	s1
i2	s2
i3	s2
i4	s1

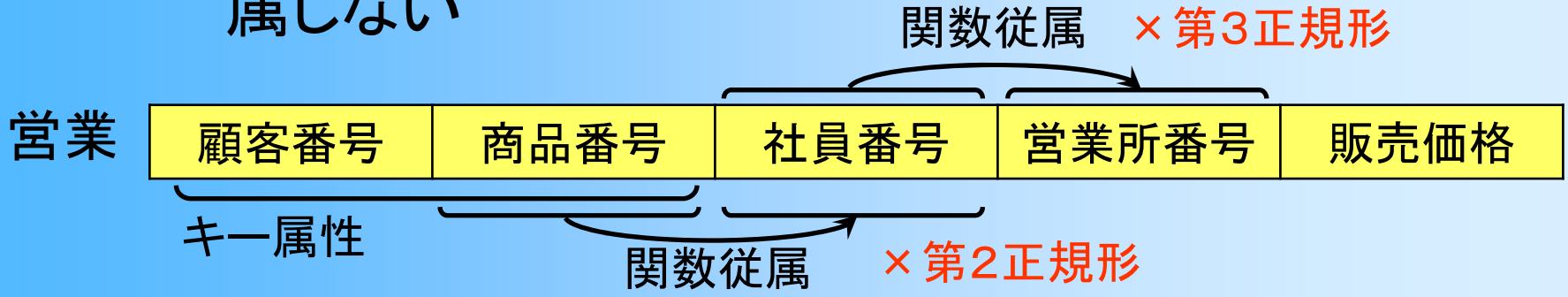
社員所属

社員番号	営業所番号
s1	o1
s2	o2

社員番号 と 営業所番号 で
1つのリレーションとする

ここまでの正規形のまとめ

- 第１正規形
  - 各属性は単一の値でなければならない
- 第２正規形
  - 候補キー以外の属性は、候補キーの部分属性に関数従属しない（キー属性が複数属性の組であるときのみ）
- 第３正規形
  - 候補キー以外の属性は、候補キー以外に関数従属しない



正規形と正規化

- 第２正規形
- 第３正規形
- ボイス･コッド正規形
- 第４正規形
- 第５正規形

ボイス・コッド正規形

- ボイス・コッド正規形
  - 第３正規形の条件をさらに厳しくしたもの
  - 「キー属性（の一部、素属性）が非キー属性に関数従属しない」

ボイス・コッド正規形の定義

- ボイス・コッド正規形の定義
  （教科書「リレーショナルデータベース入門」「データベースシステム」で共通）

  リレーションスキーマ RS において関数従属性 $X \rightarrow A$ が存在する時、常に X は RS の超キーである

  - キー属性（の一部、素属性）が非キー属性に関数従属しない
  - 第３正規形の条件から、２番目の条件を取り除いてより厳しくしたもの

ボイス･コッド正規形を満たさない例

- ボイス･コッド正規形を満たさない例
  - 各社員は一つの顧客のみを担当する
    - 一つの顧客を複数の社員が担当しても構わない

営業

商品番号	顧客番号	社員番号	販売価格
i1	c1	s1	100
i1	c2	s2	120
i2	c1	s1	200
i2	c2	s2	210
i3	c1	s3	250
i3	c2	s2	250

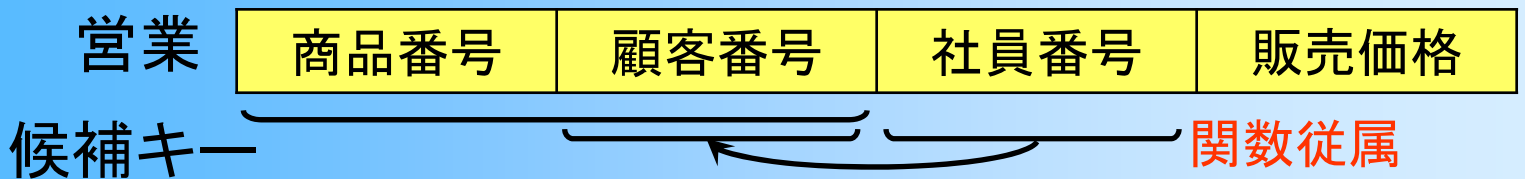
主キー

関数従属

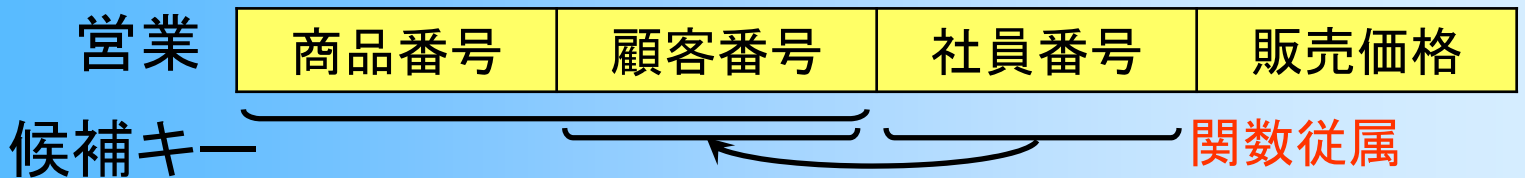
社員番号→顧客番号

ボイス・コッド正規形を満たさない例

- このリレーションスキーマの問題点
  - 社員番号が同じであれば、顧客番号も同じでなければならない
    - 社員番号→顧客番号 という関数従属性が存在
      - 逆の関数従属性（顧客番号→社員番号）は存在しないので、方向を間違えないように注意
  - 挿入不整合、修正不整合、削除不整合がそれぞれ起こる



ボイス･コッド正規形を満たすための分解



社員番号→顧客番号 の関数従属性に注目して分解

販売

商品番号	社員番号	販売価格
i1	s1	100
i1	s2	120
i2	s1	200
i2	s2	210
i3	s3	
i3	s2	

もとの営業から 顧客番号 を取り除く

社員担当

社員番号	顧客番号
s1	c1
s2	c3
s3	c1

分解により主キーが変化することに注意
{商品番号, 顧客番号}{商品番号, 社員番号}の
両方とも、もとのリレーションスキーマの候補キー

ボイス･コッド正規形を満たすための分解

- 関数従属性の情報が消えていることに注意
  - {商品番号, 顧客番号}→ 他の属性 という関数従属性（候補キー）が消えてしまう
    - 商品番号と顧客番号の値が同じであるデータが複数存在できるようになってしまう

商品番号	顧客番号	社員番号	販売価格
i1	c1	s1	100
i1	c1	s3	120

主キーの値が重複するため、このような複数のデータの存在は許されなかった

商品番号	社員番号	販売価格
i1	s1	100
i1	s3	120

主キーの値が重複しないため、存在が許される

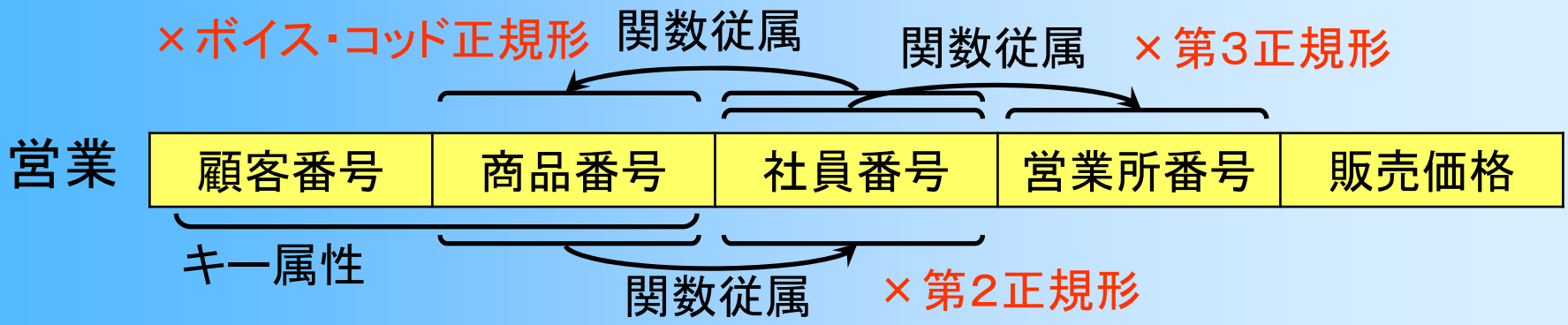
- 情報無損失分解だが、関数従属性損失となる

ここまでの正規形のまとめ

- 第１正規形
  - 各属性は単一の値でなければならない
- 第２正規形
  - 候補キー以外の属性は、候補キーの部分属性に関数従属しない（キー属性が複数属性の組であるときのみ満たさない可能性がある）
- 第３正規形
  - 候補キー以外の属性は、候補キー以外に関数従属しない
- ボイス・コッド正規形
  - 候補キーの部分属性は、非候補キーに関数従属しない
    （キー属性が複数属性の組であるときのみ満たさない可能性がある）

ここまでの正規形のまとめ

- 第２正規形
  - 候補キー以外の属性は、候補キーの部分属性に関数従属しない（キー属性が複数属性の組であるときのみ満たさない可能性がある）
- 第３正規形
  - 候補キー以外の属性は、候補キー以外に関数従属しない
- ボイス･コッド正規形
  - 候補キーの部分属性は、非候補キーに関数従属しない
    （キー属性が複数属性の組であるときのみ満たさない可能性がある）



正規形と正規化

- 第２正規形
- 第３正規形
- ボイス･コッド正規形
- 第４正規形
- 第５正規形

第４正規形

- 第４正規形
  - 多値従属性を使って定義される正規形
    - これまでの正規形は関数従属性により定義
    - 関数従属性は多値従属性の特殊な例
  - ボイス・コッド正規形の条件を関数従属性から多値従属性にまで広げたもの
  - 「キー属性（またはその一部）が非キー属性に多値従属しない」

第4正規形の定義

- 第4正規形の定義
  （教科書「リレーショナルデータベース入門」「データベースシステム」で共通）

  リレーションスキーマ RS において多値従属性 $X \rightarrow\rightarrow Y$ が存在する時、常に X は RS の超キーである

- ボイス・コッド正規形との比較
  - ボイス・コッド正規形の関数従属性（X→Y）を多値従属性（X→→Y）に広げたもの
  - より広い範囲まで含まれることになる

第４正規形でない例

- 第４正規形を満たさない例
  - 例：プロジェクトごとに社員とミーティング日が決定

プロジェクト

プロジェクト番号	社員番号	ミーティング日
p1	e1	月曜日
p1	e2	月曜日
p1	e1	木曜日
p1	e2	木曜日
p2	e1	月曜日
p2	e3	月曜日
p2	e1	金曜日
p2	e3	金曜日

候補キー

多値従属性の例(復習)

- プロジェクトごとに社員とミーティング日が決まっているものとする

プロジェクト

プロジェクト番号	社員番号	ミーティング日	
p1	e1	月曜日	t
p1	e2	月曜日	w
p1	e1	木曜日	v
p1	e2	木曜日	u

プロジェクト p1 には、社員 e1, e2 が所属
プロジェクト p1 のミーティング日は、月曜日と木曜日
プロジェクトに所属する社員とミーティング日の情報をリレーションにすると例のように4つのデータとなる

多値従属性(復習)

- 情報無損失分解の必要十分条件
- 多値従属性の定義

リレーションスキーマ RS のリレーション R が、以下の条件を満たす時、X は Y に多値従属 $\left(X \to\to Y\right)$

$$\begin{aligned}&\left(\forall t \in R\right)\left(\forall u \in R\right)\left(t\left[X\right] = u\left[X\right] \to \left(\exists v \in R\right)\left(\exists w \in R\right)\right.\\&\quad\left(\left(t\left[X\right] = u\left[X\right] = v\left[X\right] = w\left[X\right]\right) \wedge\right.\\&\qquad t\left[Y\right] = v\left[Y\right] \wedge t\left[RS - XY\right] = w\left[RS - XY\right] \wedge\\&\qquad\left.\left.u\left[Y\right] = w\left[Y\right] \wedge u\left[RS - XY\right] = v\left[RS - XY\right]\right)\right)\end{aligned}$$

多値従属性(復習)

- 情報無損失分解の必要十分条件
- 多値従属性の定義

リレーションスキーマ RS のリレーション R が、以下の条件を満たす時、X は Y に多値従属 $\left(X \rightarrow\rightarrow Y\right)$

t(x, y1, z1), u(x, y2, z2) というタプルが存在すれば 、必ず v(x, y1, z2), w(x, y2, z1) のようなタプルも存在する

多値従属性の例(復習)

- プロジェクトごとに社員とミーティング日が決まっているものとする

プロジェクト

プロジェクト番号	社員番号	ミーティング日
p1	e1	月曜日
p1	e2	月曜日
p1	e1	木曜日
p1	e2	木曜日
p2	e1	月曜日
p2	e3	月曜日
p2	e1	金曜日
p2	e3	金曜日

多値従属性の例(復習)

- プロジェクトごとに社員とミーティング日が決まっているものとする

プロジェクト

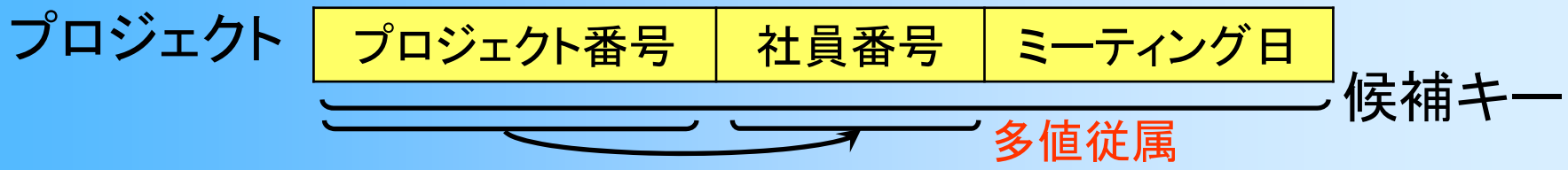
プロジェクト番号	社員番号	ミーティング日	
p1	e1	月曜日	t
p1	e2	月曜日	w
p1	e1	木曜日	v
p1	e2	木曜日	u
	e1	月曜日	
	e3	月曜日	

$t(x, y_1, z_1)$, $u(x, y_2, z_2)$ というタプルが存在すれば、必ず $v(x, y_1, z_2)$, $w(x, y_2, z_1)$ のようなタプルも存在する

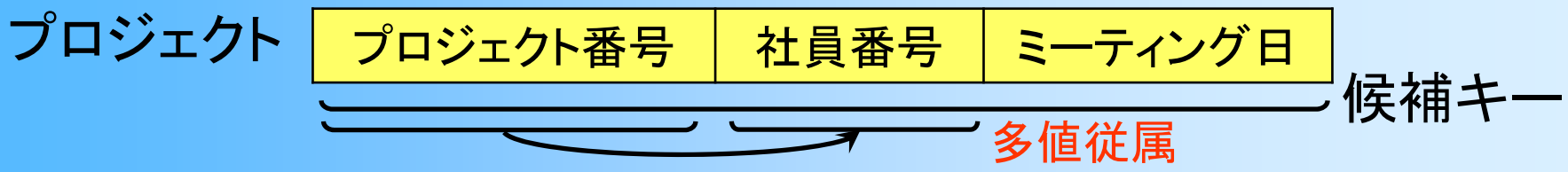
X, Y は複数の属性の組でも良く、Z＝RS－XY であることに注意

第４正規形を満たす分解

- 第４正規形を満たさない例
  - プロジェクト番号 →→ 社員番号 | ミーティング日 の多値従属性が存在
  - プロジェクト番号は、もとのリレーションの超キーではないため、第４正規形の定義を満たさない



第４正規形を満たす分解



- 多値従属性に注目して分割

  プロジェクト番号 →→ 社員番号 | ミーティング日

参加社員

プロジェクト番号	社員番号
p1	e1
p1	e2
p2	e1
p2	e3

ミーティング日

プロジェクト番号	ミーティング日
p1	月曜日
p1	木曜日
p2	月曜日
p2	金曜日

正規形と正規化

- 第２正規形
- 第３正規形
- ボイス･コッド正規形
- 第４正規形
- 第５正規形

第５正規形

- 結合従属性

  $*\left(RS_1, \cdots, RS_n\right)$

  – リレーション RS が複数のリレーション RS_i に情報無損失分解できる
  – 多値従属性の一般形
    - n=2、2つのリレーションに分解できるときには、多値従属性があると言える

第５正規形

- 第５正規形の定義
  （教科書「リレーショナルデータベース入門」「データベースシステム」で共通）
  - リレーション RS に、結合従属性 $*\left(RS_1,\cdots,RS_n\right)$ が存在する時、各 RS_i は RS の超キーである
  - RS_i が超キーとならない結合従属性があれば、リレーション RS は第５正規形を満たさない

第５正規形を満たさない例

- 第５正規形を満たさない例
  - 工場 f は、部品 p の供給を、f と契約している業者 s で p が供給可能な全ての業者 s から受ける

部品供給

工場番号	部品番号	業者番号
*f*1	p1	s1
*f*1	p2	s1
*f*1	p2	s2
*f*1	p3	s2
*f*2	p2	s2
*f*2	p3	s3

候補キー

第５正規形を満たさない例

- 第５正規形を満たさない例
  - 工場 *f* は、部品 *p* の供給を、*f* と契約している業者 *s* で *p* が供給可能な全ての業者 *s* から受ける

① 工場 *f1* が部品 *p2* の供給を受けていて、

② 工場 *f1* が業者 *s2* から別の部品の供給を受けていて、

部品供給

工場番号	部品番号	業者番号
*f*1	p1	s1
*f*1	p2	s1
*f*1	p2	s2
*f*1	p3	s2
*f*2	p2	s2
*f*2	p3	s3

候補キー

④ 工場 *f1* は業者 *s2* から部品 *p2* の供給を必ず受けるとする

③ 業者 *s2* が部品 *p2* を別の工場に供給していれば、

第５正規形を満たさない例

- このスキーマには、以下の結合従属性が存在

$$*\left(\{\text{工場番号, 部品番号}\},\{\text{工場番号, 業者番号}\},\{\text{部品番号, 業者番号}\}\right)$$

– 3つのリレーションに情報無損失分解できる

製造部品

工場番号	部品番号
*f*1	p1
*f*1	p2
*f*1	p3
*f*2	p2
*f*2	p3

工場担当

工場番号	業者番号
*f*1	s1
*f*1	s2
*f*2	s2
*f*2	s3

部品担当

部品番号	業者番号
p1	s1
p2	s1
p2	s2
p3	s2
p3	s3

第５正規形を満たさない例

- 第５正規形を満たさない

$$*\left(\{\text{工場番号}, \text{部品番号}\}, \{\text{工場番号}, \text{業者番号}\}, \{\text{部品番号}, \text{業者番号}\}\right)$$

  - 分解後の各リレーション RS_i は、もとのリレーションの超キーではないことに注意

部品供給

工場番号	部品番号	業者番号
*f*1	p1	s1
*f*1	p2	s1
*f*1	p2	s2
*f*1	p3	s2
*f*2	p2	s2
*f*2	p3	s3

候補キー

第５正規形を満たすための分解

- 第５正規形を満たすように分解

製造部品

工場番号	部品番号
*f*1	p1
*f*1	p2
*f*1	p3
*f*2	p2
*f*2	p3

工場担当

工場番号	業者番号
*f*1	s1
*f*1	s2
*f*2	s2
*f*2	s3

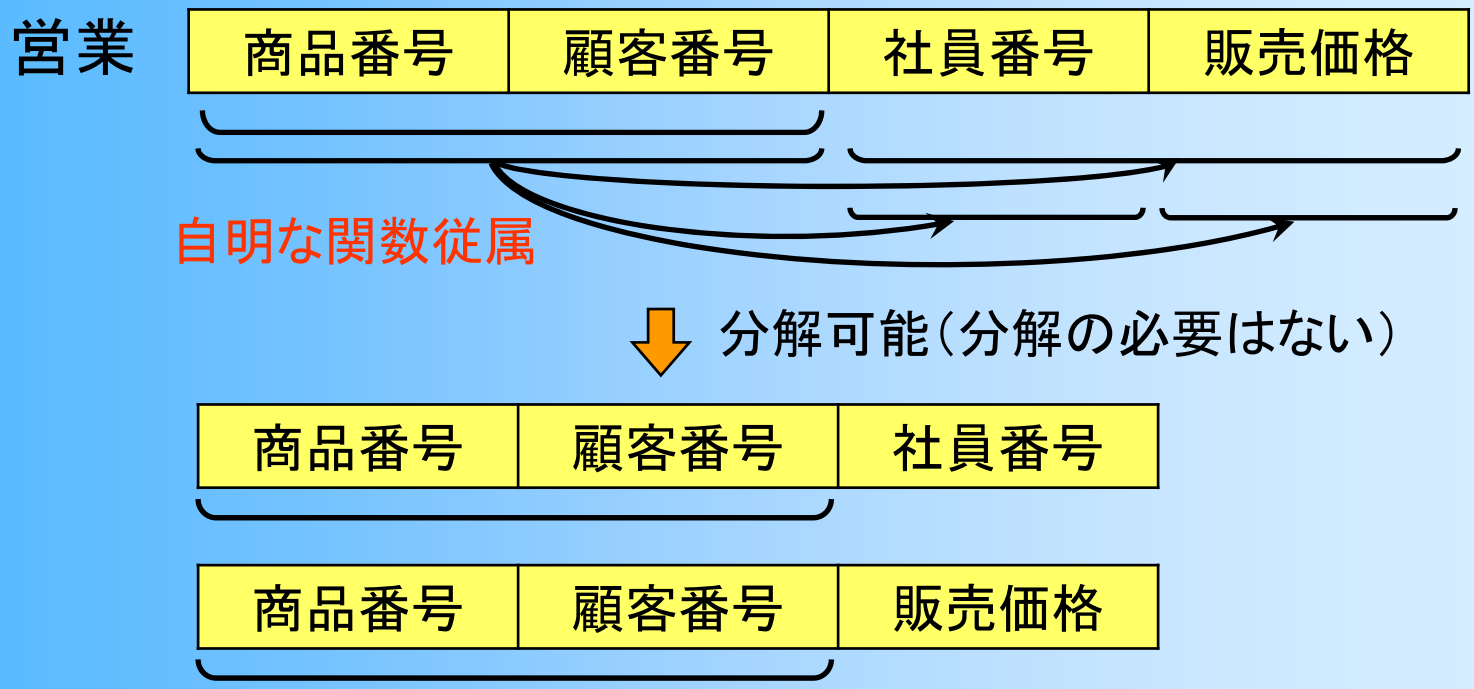
部品担当

部品番号	業者番号
p1	s1
p2	s1
p2	s2
p3	s2
p3	s3

２つに分解しただけでは結合してももとに戻らないことに注意（情報損失分解）
この場合は、必ず３つのリレーションに分割する必要がある

第５正規形を満たす例

- 不必要に分解する必要はない
  - 分解後のリレーションは、もとのスキーマの超キーなので、分解しなくても第５正規形を満たす

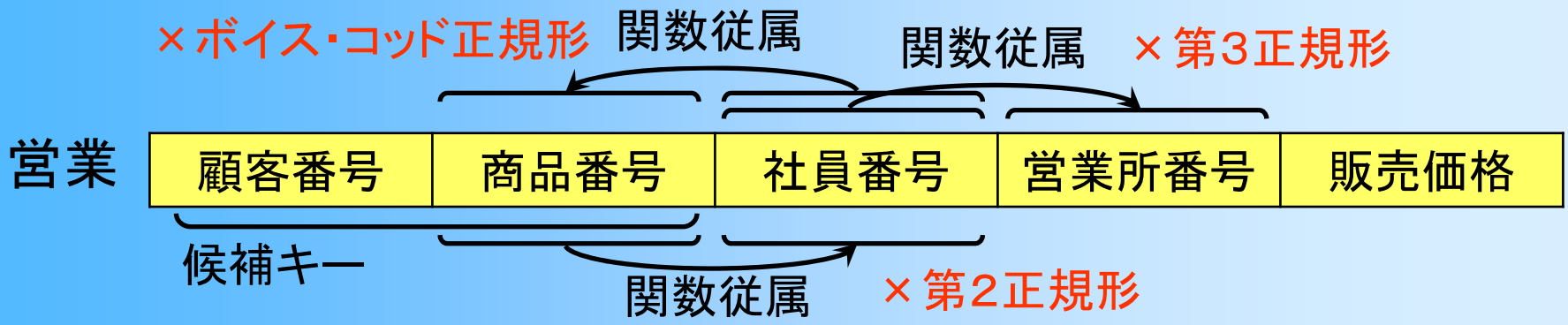


正規形のまとめ

- 第２正規形
- 第３正規形
- ボイス･コッド正規形
- 第４正規形
- 第５正規形

正規形の条件のまとめ(1)

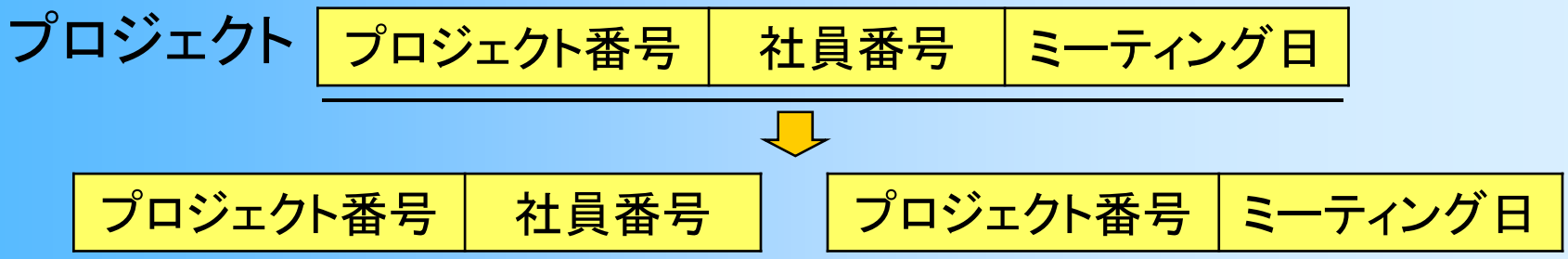
- 第2正規形
  - 候補キー以外の属性は、候補キーの部分属性に関数従属しない(キー属性が複数属性の組であるときのみ満たさない可能性がある)
- 第3正規形
  - 候補キー以外の属性は、候補キー以外に関数従属しない
- ボイス・コッド正規形
  - 候補キーの部分属性は、非候補キーに関数従属しない
    (キー属性が複数属性の組であるときのみ満たさない可能性がある)



正規形の条件のまとめ(2)

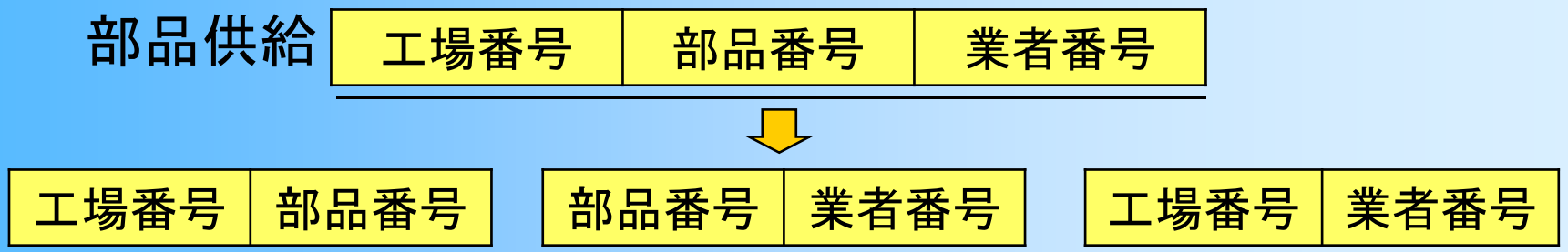
- 第4正規形
  - 自明でない多値従属が存在しない

多値従属性　プロジェクト番号 →→ 社員番号 | ミーティング日



- 第5正規形
  - 自明でない結合従属性が存在しない

結合従属性　＊({工場番号, 部品番号}, {部品番号, 業者番号}, {工場番号, 業者番号})

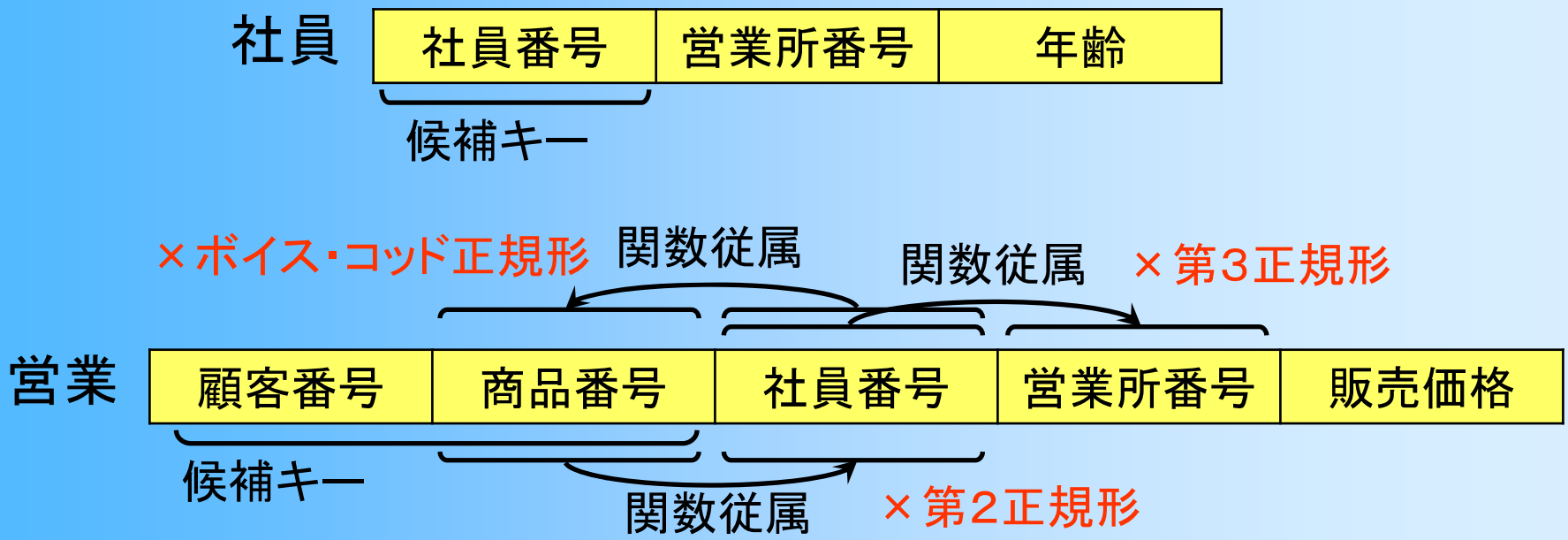


正規形の判定方法(1)

- 関数従属性を正しく書き出す(重要)
- 関数従属性にもとづき、正規形の定義に従って、正規形を判定(基本的にはこれだけ)
- 第2正規形から順番に判定していく
  - 途中の正規形を飛ばさない
- 第4正規形(多値従属性)、第5正規形(結合従属性)は、別に考える
  - ほとんどの場合、これらの正規形は満たすので、あまり心配する必要は無い

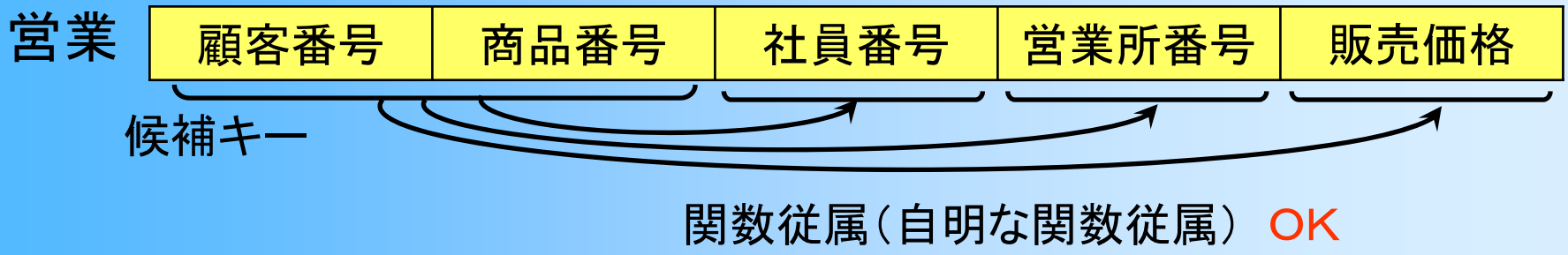
正規形の判定方法(2)

- 1つの属性のみで候補キーとなっていれば、自動的に、第2正規形、ボイスコッド・正規形は満たす
  - 候補キーの部分属性が存在しないため



正規形の判定方法(3)

- 自明でない関数従属性(候補キー属性 → 非キー属性 以外の関数従属性)がなければ、第2正規形～ボイス・コッド正規形は満たす
  - 後は、自明でない多値従属性(第4正規形)や、自明でない結合従属性(第5正規形)がないかを確認する



まとめ

- 前回の復習
  - 正規化の必要性
  - 関数従属性と多値従属性
- 正規形と正規化
  - 第２正規形
  - 第３正規形
  - ボイス･コッド正規形
  - 第４正規形
  - 第５正規形

次回予告

- リレーションスキーマの設計
  - 概念設計と論理設計
  - 実体関連モデルによるスキーマの設計
  - 正規化によるスキーマの設計