データベース

第9回　リレーションスキーマの設計(3)

九州工業大学 情報工学部　尾下真樹

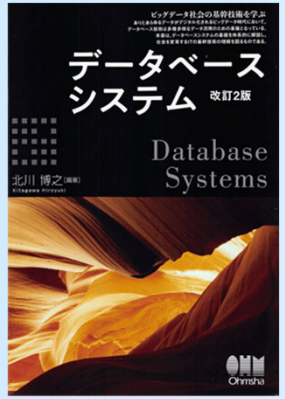
今日の内容

- 前回の復習
  - 正規形と正規化
    - 第2正規形～第５正規形
  - 正規形のまとめ
- 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
- レポート課題（前半）

教科書・参考書

- 「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（3,200円）
  - 1章（1.3、1.5～1.6）

- 「データベースシステム 改訂2版」
  北川 博之 著、オーム社（3,200円）
  - 7章（7.1）



前回の復習

正規形

- 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

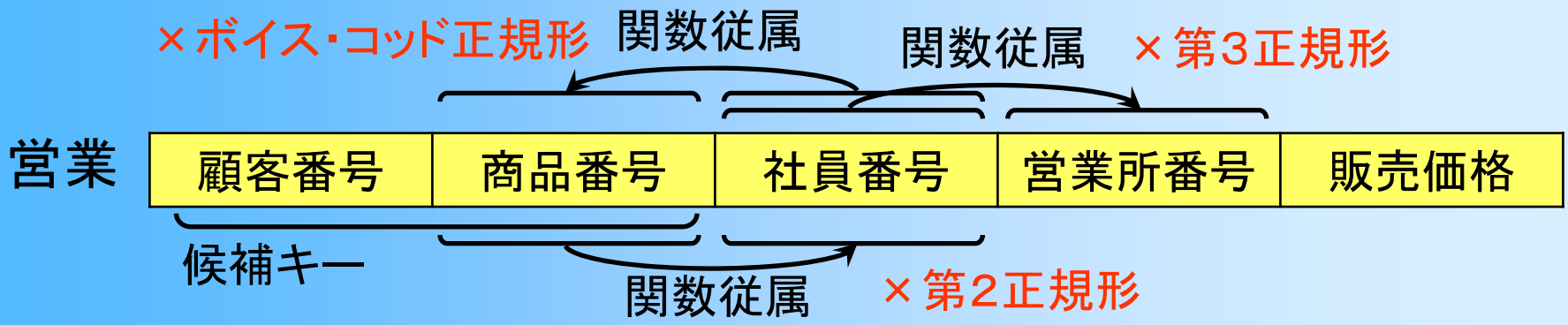
- 関数従属性や多値従属性にもとづいて、リレーションが正規形を満たすかどうかを判定
- 正規形を満たさないリレーションがあれば、正規形を満たすように、リレーションを分解（＝正規化）
- 段階的に正規化を行っていき、最終的には第５正規形まで満たすようにする

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が存在すれるときに、必ず全ての Y と（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になる
    - Yが常に1種類のみ存在するもの

正規形の条件のまとめ(1)

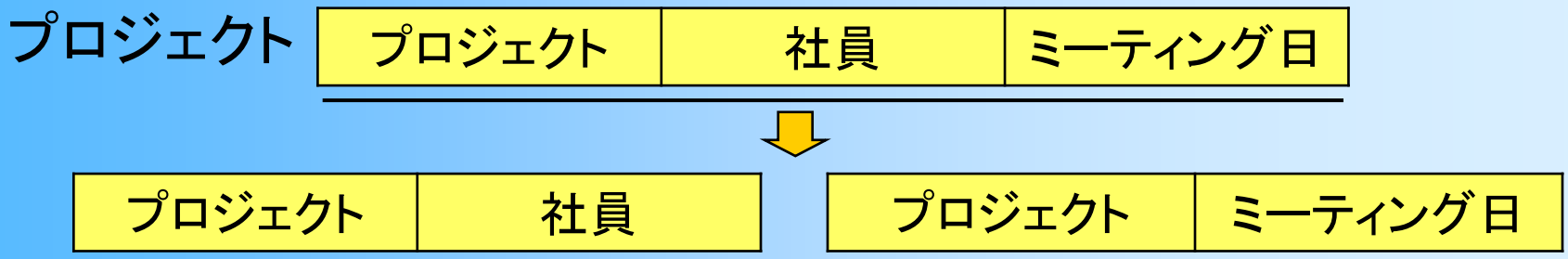
- 第2正規形
  - 候補キー以外の属性は、候補キーの部分属性に関数従属しない(キー属性が複数属性の組であるときのみ満たさない可能性がある)
- 第3正規形
  - 候補キー以外の属性は、候補キー以外に関数従属しない
- ボイス・コッド正規形
  - 候補キーの部分属性は、非候補キーに関数従属しない
    (キー属性が複数属性の組であるときのみ満たさない可能性がある)



正規形の条件のまとめ(2)

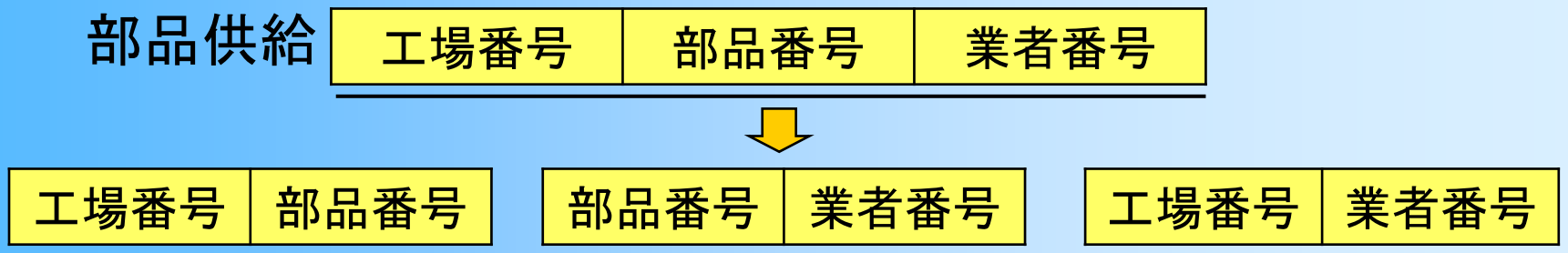
- 第4正規形
  - 自明でない多値従属が存在しない

多値従属性　プロジェクト →→ 社員 | 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})

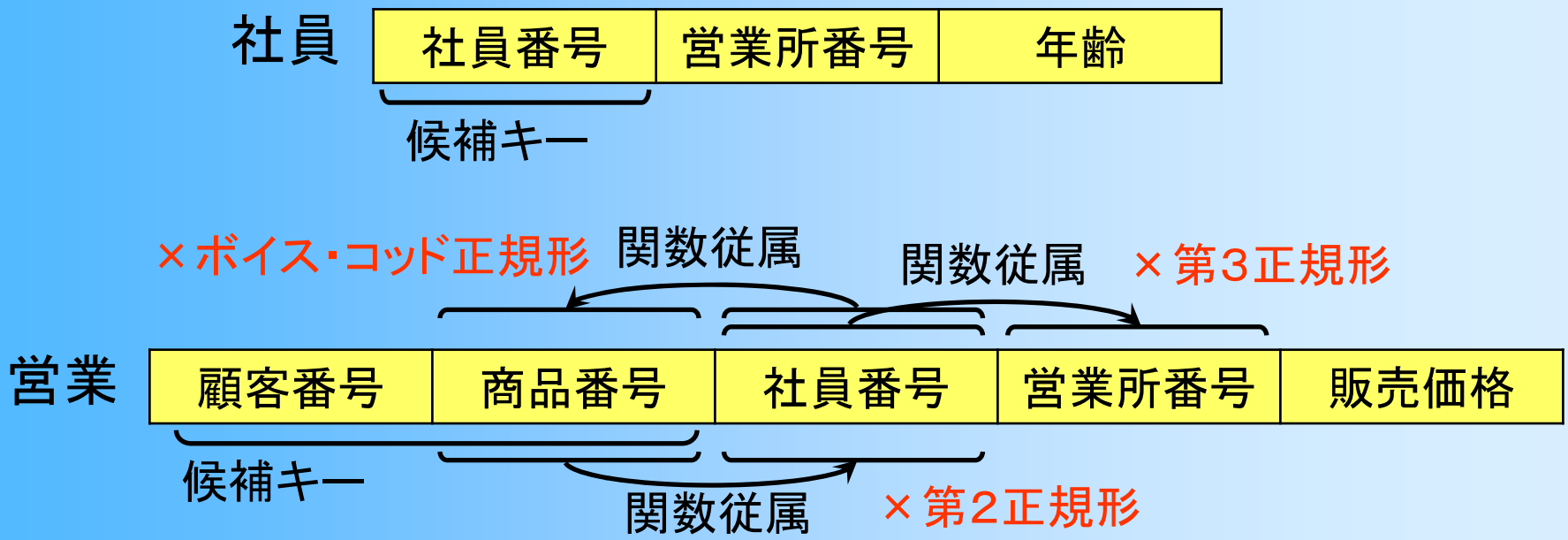


正規形の判定方法(1)

- 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

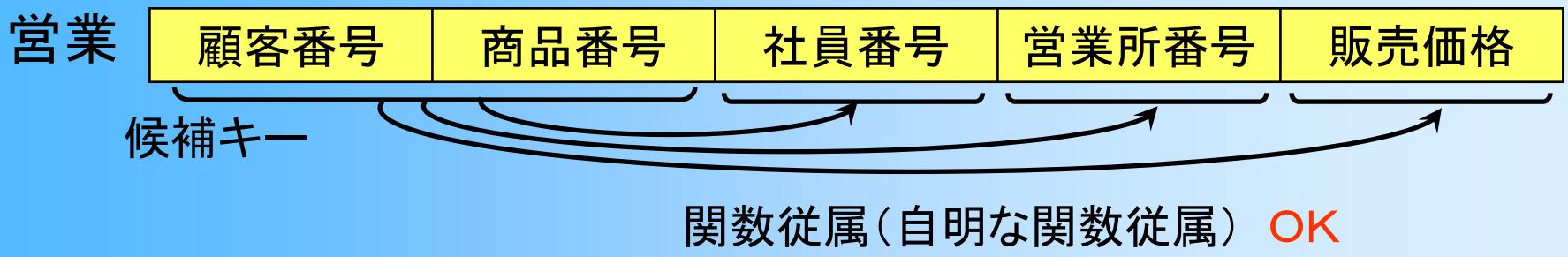
正規形の判定方法(2)

- 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



演習問題(1)

どのような関数従属性 or 多値従属性が存在するか？
どの正規形を満たさないか？

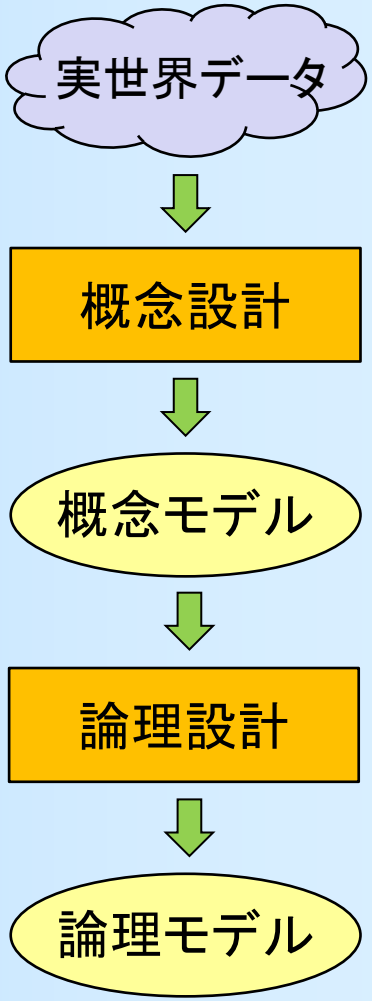
営業(商品番号, 顧客番号, 社員番号, 販売価格)

各社員は、複数の顧客に対する営業を担当する。
一つの顧客に対して、複数の社員が営業を担当する。
商品ごとに一人の担当社員が決まっている。
一人の社員は、複数の商品の営業を担当する。
同じ商品であっても、顧客によって価格は異なる。

リレーションスキーマの設計

リレーションスキーマの設計

- データベースシステムを利用するためには、データベースに格納したい現実のデータを、データベーシステムが提供するデータモデルを使って記述する必要がある
  - 概念設計
    - 現実のデータの概念を整理
  - 論理設計
    - 具体的なスキーマを決定



リレーションスキーマの設計

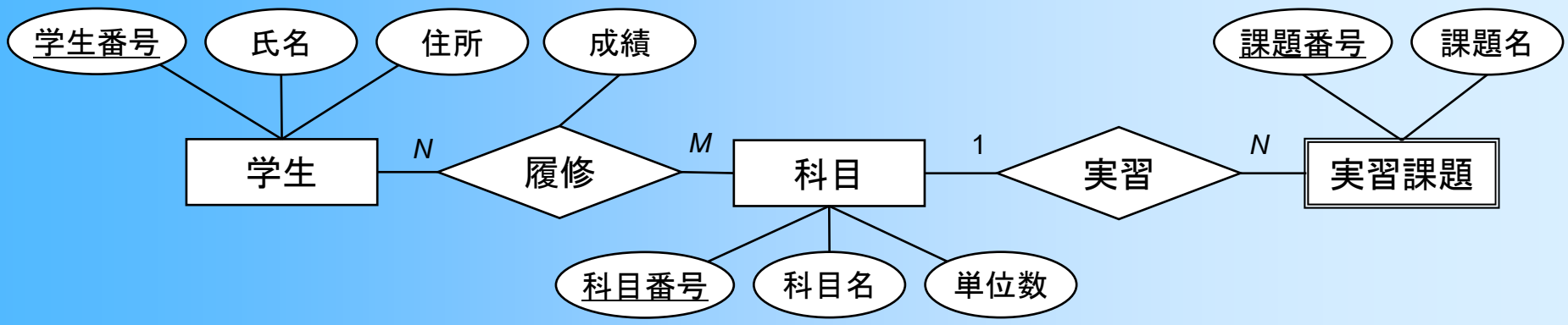
- 概念設計の方法
  - 実体関連モデルを用いる方法
- 論理設計の方法
  - 実体関連モデルからスキーマを決定する方法
  - 仮のスキーマを正規化していく方法

実体関連モデル

- 概念設計を行うときのひとつの方法
  - 実体関連モデルは、概念設計の考え方のひとつなので、どのデータモデルにも適用できる
    - 第2回の講義で紹介した各種データモデルとは別なので混乱しないこと
  - 実体と関連
    - 実体
      - ひとつの実体をその属性によって表したもの
    - 関連
      - 複数の実体間の関連を表すもの
      - 関連にも属性を持たせることが可能

実体関連モデルの記述方法(1)

- 実体関連図(ER図)
  - 実体関連モデルを使ってモデル化した概念を図に表したもの
    - 実体は四角、関連はひし形、属性は丸、キー属性はアンダーラインで表されている

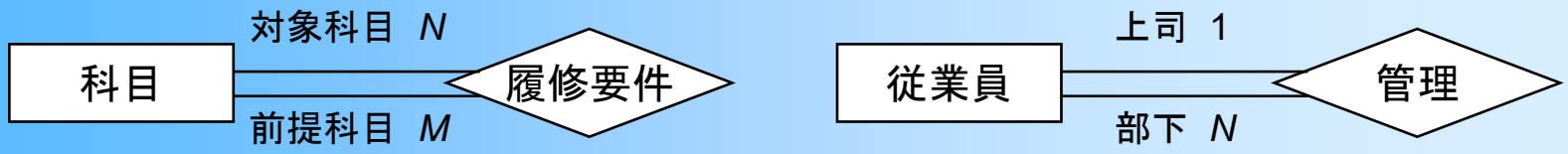


実体関連モデルの記述方法(2)

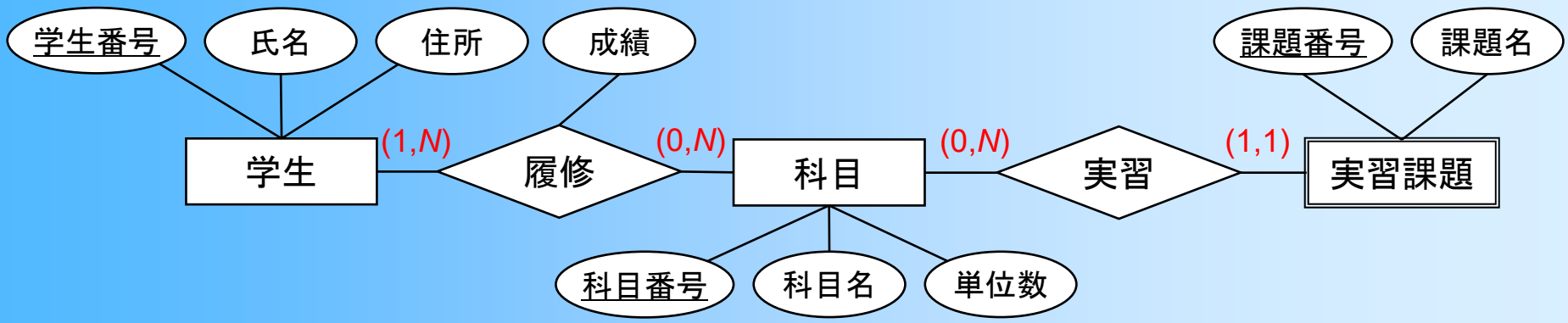
- 参加制約
  - 関連の整合性を保つための制約
  - 1対1、1対N(1対多)、N対M(多対多)の区別
  - 同一の実体・関連間で複数の関与もありうる
  - 関連が全くなくてもいいか、最低ひとつはいるか
- キー制約
  - その属性値によって実体を一意に特定できるような属性に対する制約
  - キー制約を持つ属性が複数あっても良い
    - 主キーと候補キー

実体関連モデルの記述方法(3)

- 同じ実体に複数の関与がある関連の例

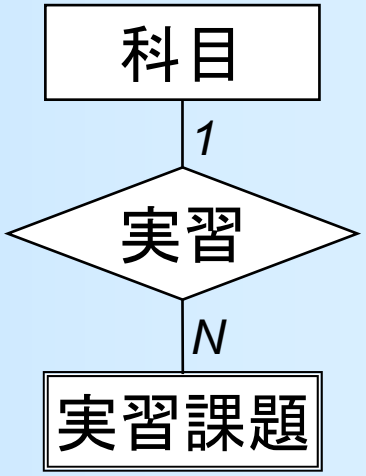


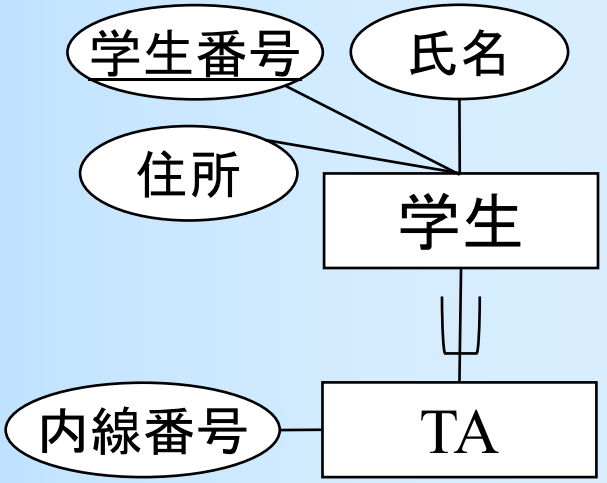
- 参加制約を明確にした実体関連図の例
  - 関連の範囲(0 or 1～1 or *N*)を明示した書き方



実体関連モデルの記述方法(4)

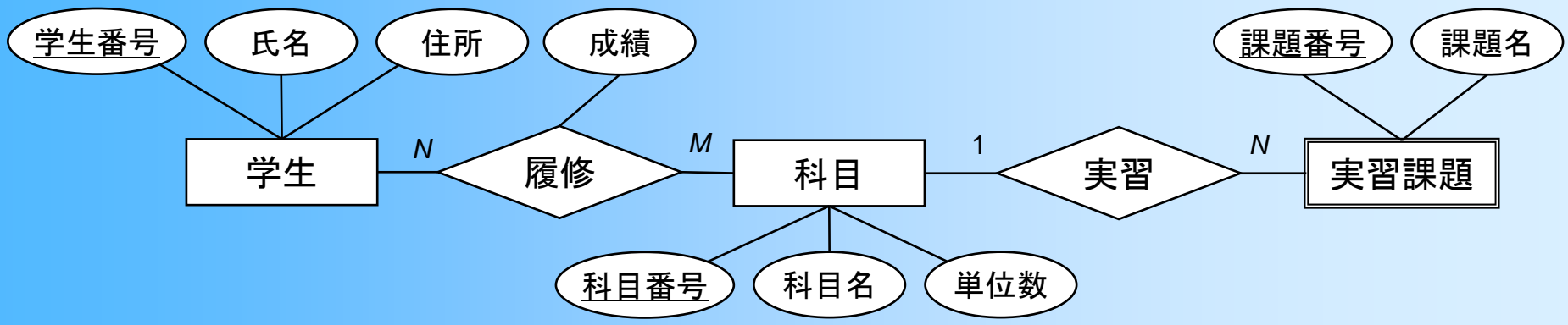
- 弱実体
  - ある実体に付属する実体
  - 付属先の実体からの参照を持つ
- 汎化階層
  - 抽象的な実体から、具体的な実体に派生
  - 派生元の実体の属性＋派生した実体の属性を持つ
  - オブジェクト指向の考え方





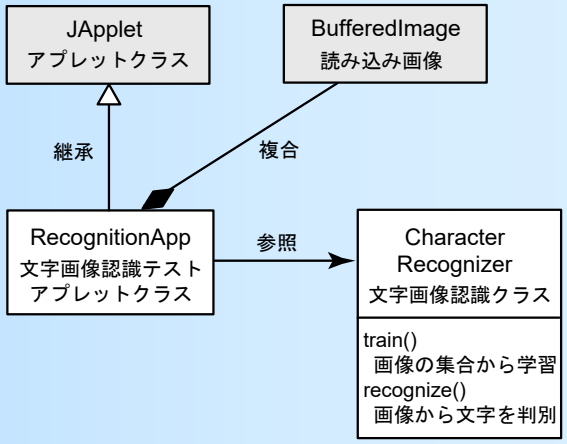
実体関連図の書き方

- 実体や関連を書き出していく
  - 実体は四角、関連はひし形で
  - 実体・関連に適宜、属性の情報を加える
  - 関連と実体の対応関係（単数or複数など）に注意



実体関連図の応用

- プログラム開発でも、同様の考え方でクラス設計（概念設計・論理設計）を行う
  - オブジェクト指向プログラミングの設計でも、実体や関連に注目してクラスを定義
  - クラス図
    - 実体関連図と同様、各クラスと、クラスが持つ属性やメソッド、クラス間の関連を図に記述
    - Unified Modeling Language（UML）に従って記述



クラス図の例

リレーションスキーマの設計

- 概念設計の方法
  - 実体関連モデルを用いる方法
- 論理設計の方法
  - 実体関連モデルからスキーマを決定する方法
  - 仮のスキーマを正規化していく方法

論理設計の方法

1. 実体関連モデルからスキーマを作る方法
2. スキーマを正規化していく方法
   - 正規形を満たすように、スキーマを分解
   - 一般に、正しく実体関連モデルが設計されていれば、正規形を満たすスキーマが作られるので、正規化の必要はない場合が多い
   - 実体関連モデルを用いず、正規化のみでスキーマの設計を行なうこともできる

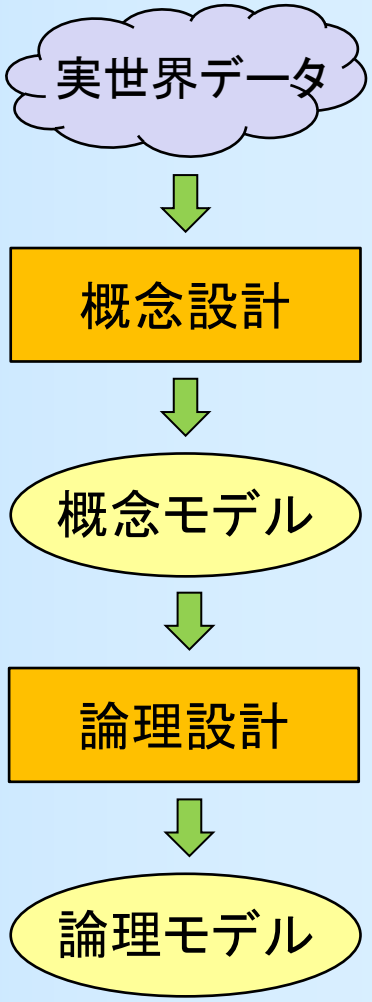
- 組み合わせて（あるいは一方のみを）使用
- 両方の方法を使えることが必要

リレーションスキーマの設計

- 概念設計の方法
  - 実体関連モデルを用いる方法
- 論理設計の方法
  - 実体関連モデルからスキーマを決定する方法
  - 仮のスキーマを正規化していく方法

実体関連モデルからのスキーマの設計

- スキーマ設計の手順
  - 概念設計
  - 論理設計
- 実体関連モデル
  - 概念設計の方法
  - データベースに格納すべき情報を、実体と関連に分けて整理する
- 実体関連モデルから、スキーマの論理設計を導出



実体からリレーションを導出

- 実体集合
  - 1つの実体 E からリレーション R を定義
  - 実体の属性はリレーションの属性に
- 弱実体集合（ある実体 E’ に付属する実体）
  - 1つの弱実体 E からリレーション R を定義
  - オーナ実体の主キーを参照として主キーに追加
- 汎化階層（ある実体 E’ を派生させた実体）
  - 1つの汎化階層 E からリレーション R を定義
  - 上位実体の主キーを参照として主キーに追加

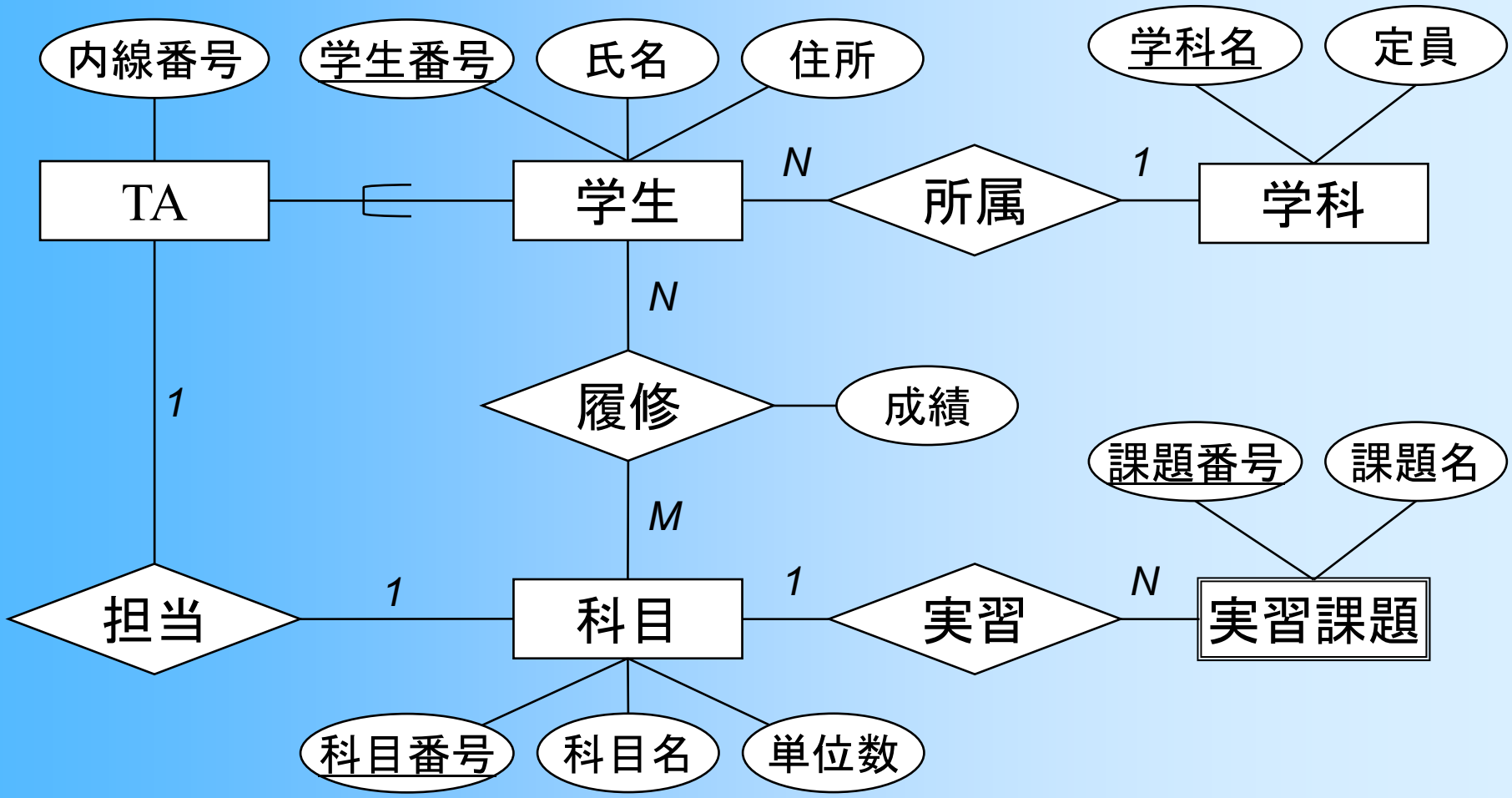
関連からリレーションを導出

- 2次の関連

  - 1対1の場合
    - いずれか片方のリレーションに、もう一方の主キーを参照として追加（属性を追加）
  - 1対多（1対N）の場合
    - 1対多の多の側（Nの側）のリレーションに、もう一方の主キーを参照として追加（属性を追加）
  - 多対多（N対M）の場合
    - 両方の実体の主キーを参照するリレーションを作成
- 3次の関連
  - 2次のN対Mの場合と同様（リレーションを作成）

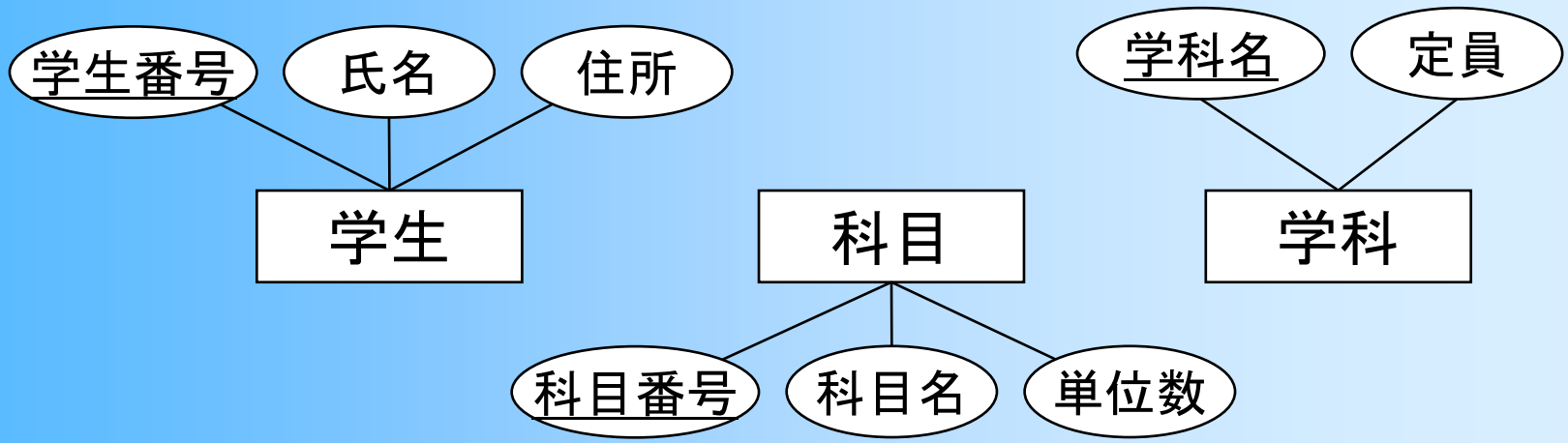
リレーション導出の例

- 具体例からリレーションスキーマを作成



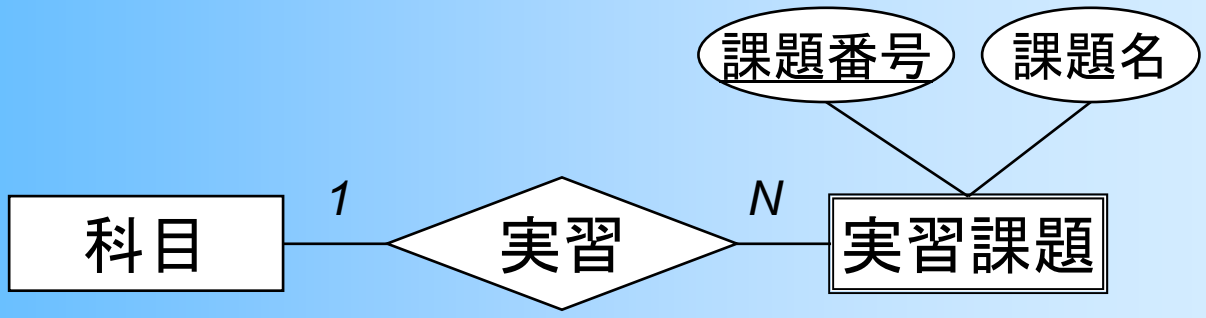
リレーション導出の例(1)

1. 実体集合からリレーションを導出
   - 学生(学生番号, 氏名, 住所)
   - 科目(科目番号, 科目名, 単位数)
   - 学科(学科名, 定員)



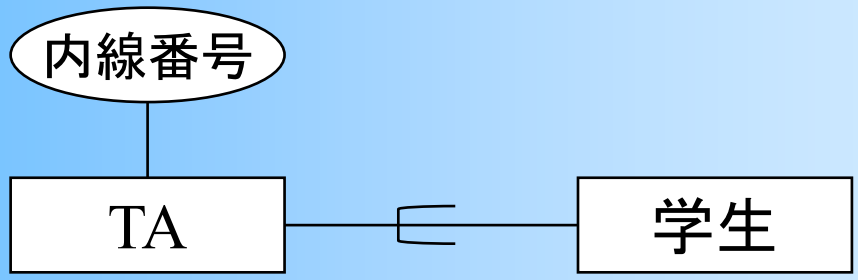
リレーション導出の例(2)

2. 弱実体からリレーションを導出
   - 実習課題(科目番号, 課題番号, 課題名)
     - 元の実体(科目)の主キー属性(科目番号)を追加



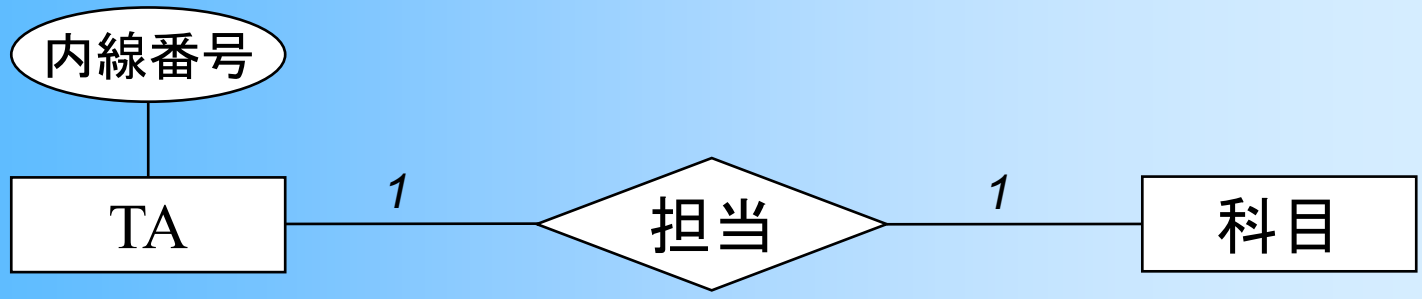
リレーション導出の例(3)

3. 汎化階層からリレーションを導出
   - TA(学生番号, 内線番号)
     - 上位実体(学生)の主キー(学生番号)を参照として主キーに追加



リレーション導出の例(4)

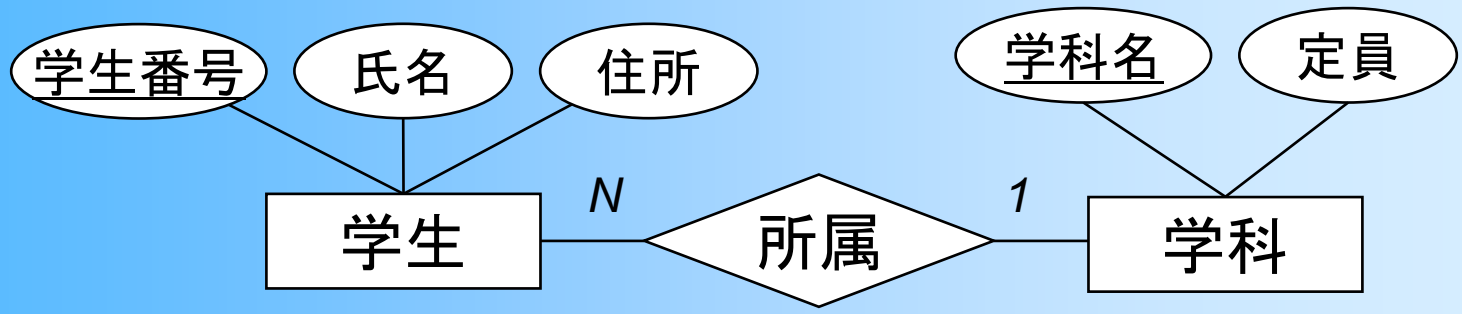
4. 1対1の関連集合からリレーションを修正
   - TA(学生番号, 内線番号) →
     TA(学生番号, 内線番号, 科目番号)
     - 担当の関連にもとづき、TAに科目の主キー属性(科目番号)を追加
       - 代わりに、科目にTAの主キー属性を追加しても良い



リレーション導出の例(5)

5. 1対多の関連集合からリレーションを修正

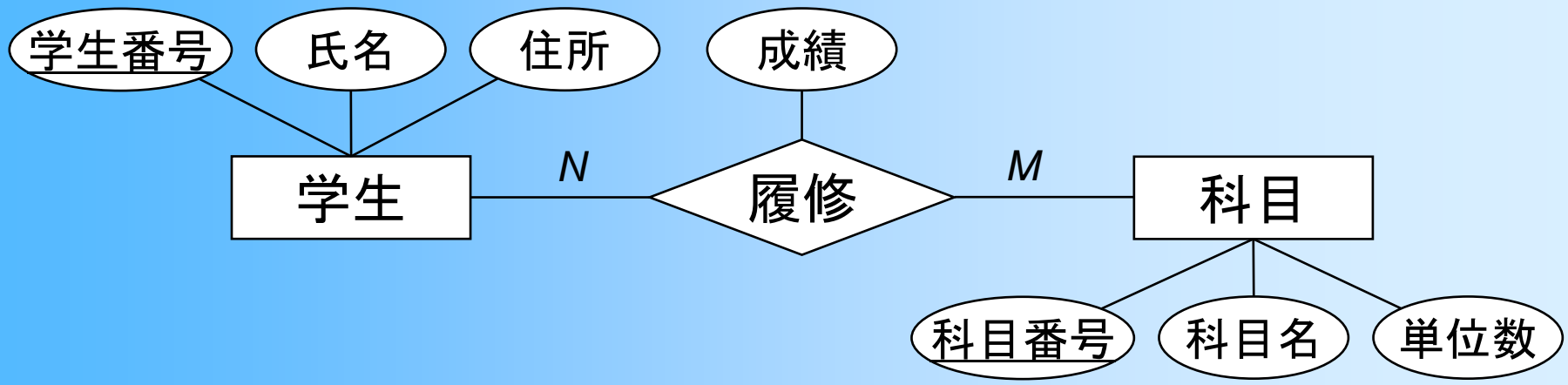
- 学生(学生番号, 氏名, 住所) →
  学生(学生番号, 氏名, 住所, 学科名)
  - 所属の関連にもとづき、学生に学科の主キー属性(学科名)を追加



リレーション導出の例(6)

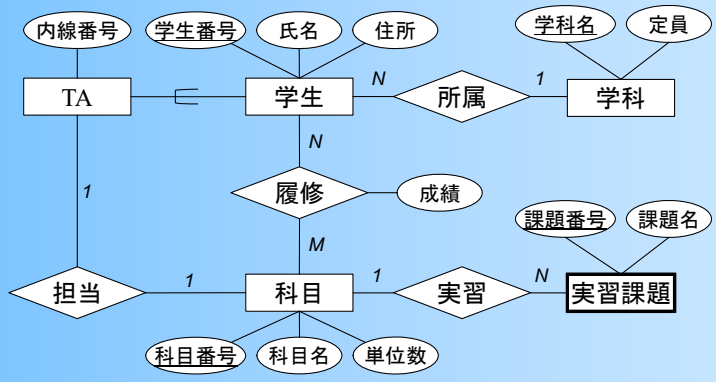
6. 多対多の関連集合からリレーションを導出

- 履修(科目番号, 学生番号, 成績)
  - 科目と学生の両方を参照するための属性(両方のリレーションの主キー、科目番号と学生番号)を、履修の主キーとする



リレーション導出の例

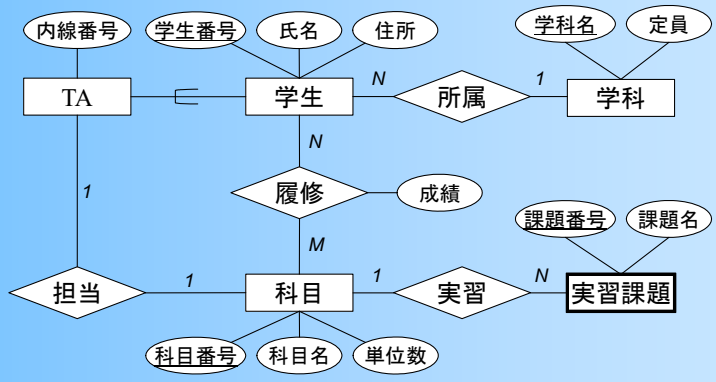
- 最終的に得られたスキーマ（全ての正規形を満たす）
  - 学生（学生番号, 氏名, 住所, 学科名）
  - 科目（科目番号, 科目名, 単位数）
  - 学科（学科名, 定員）
  - 実習課題（科目番号, 課題番号, 課題名）
  - TA（学生番号, 内線番号, 科目番号）
  - 履修（科目番号, 学生番号, 成績）



リレーション導出の例

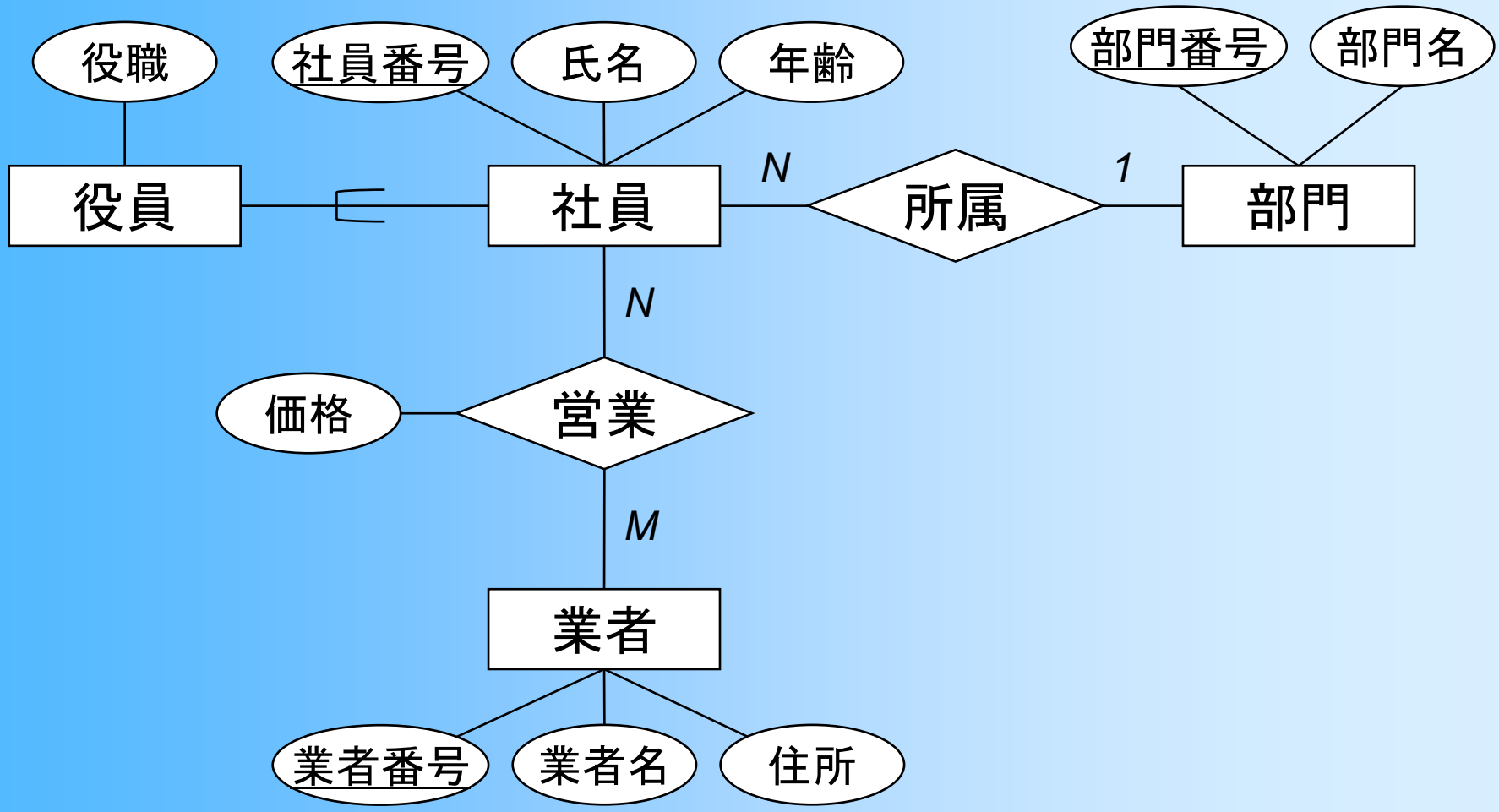
- 最終的に得られたスキーマ（全ての正規形を満たす）
  - 学生（学生番号, 氏名, 住所, 学科名）
  - 科目（科目番号, 科目名, 単位数）
  - 学科（学科名, 定員）
  - 実習課題（科目番号, 課題番号, 課題名）
  - TA（学生番号, 内線番号, 科目番号）
  - 履修（科目番号, 学生番号, 成績）

赤字の属性は、外部キー属性（他のインスタンスを参照する属性）



演習問題(2)

実体関連図からリレーションスキーマを作成せよ



リレーションスキーマの設計

- 概念設計の方法
  - 実体関連モデルを用いる方法
- 論理設計の方法
  - 実体関連モデルからスキーマを決定する方法
  - 仮のスキーマを正規化していく方法

スキーマを正規化していく方法

1. 必要な属性を書き出して、仮のスキーマを作成する
2. 関数従属性を書き出す
3. 正規形を満たすように、スキーマを分解していく

具体例(1)

- 初期スキーマの作成
  - 履修(学生番号、科目番号、氏名、所属学部、所属学科、住所、科目名、単位数、成績)
- このリレーションに格納されるデータの条件から、候補キーや関数従属性を書き出す
  - 自明でない関数従属性を書き出す
    - 自明な関数従属性(候補キー全体→他の属性)は、書き出しても、書き出さなくても、どちらでも構わない(正規化には影響しない)
  - 自明でない多値従属性・結合従属性も書き出す

具体例(2)

- 格納されるデータの条件
  1. 各学生の各科目の履修は1つしか存在しない
     - {学生番号, 科目番号} が候補キー(主キー)となる
  2. 学生番号から、氏名、所属学部、所属学科、住所が決まる
     - 学生番号 → 氏名、所属学部、所属学科、住所
  3. 科目番号から、科目名、単位数が決まる
     - 科目番号 → 科目名、単位数
  4. 所属学科から、所属学部が決まる
     - 所属学科 → 所属学部

具体例(3)

- 初期スキーマ
  - 履修(学生番号、科目番号、氏名、所属学部、所属学科、住所、科目名、単位数、成績)
- 関数従属性
  - 学生番号 → 氏名、所属学部、所属学科、住所
  - 科目番号 → 科目名、単位数
  - 所属学科 → 所属学部

  ※ 自明な関数従属性(候補キー全体→他の属性)は書き出しても、書き出さなくても、構わない
    - 例:学生番号、科目番号 → 成績

具体例(4)

- スキーマと関数従属性
  - 履修(学生番号、科目番号、氏名、所属学部、所属学科、住所、科目名、単位数、成績)
  - 学生番号 →氏名、所属学部、所属学科、住所
  - 科目番号 →科目名、単位数
  - 所属学科 → 所属学部
- 履修リレーションは、候補キーの一部の属性から 候補キー以外の属性 への関数従属性があるため、第2正規形を満たさない

具体例(5)

- 正規形を満たすように分解
  - 履修(学生番号、科目番号、氏名、所属学部、所属学科、住所、科目名、単位数、成績)

  - 履修(学生番号、科目番号、成績)
  - 学生(学生番号、氏名、所属学部、所属学科、住所)
  - 科目(科目番号、科目名、単位数)

具体例(6)

- 分解後のスキーマと関数従属性
  - 履修(学生番号、科目番号、成績)
  - 学生(学生番号、氏名、所属学部、所属学科、住所)
  - 科目(科目番号、科目名、単位数)
  - 所属学科 → 所属学部
- 学生リレーションは、候補キー以外の属性から 候補キー以外の属性 への関数従属性があるため、第3正規形を満たさない

具体例(7)

- 正規形を満たすように分解
  - 学生(学生番号、氏名、所属学部、所属学科、住所)

  - 学生(学生番号、氏名、所属学科、住所)
  - 学科(所属学科、所属学部)

具体例(8)

- 分解後のスキーマ
  - 履修(学生番号、科目番号、成績)
  - 学生(学生番号、氏名、所属学科、住所)
  - 学科(所属学科、所属学部)
  - 科目(科目番号、科目名、単位数)
- このスキーマは他の正規形も全て満たす
  - ボイス・コッド正規形(他の関数従属性はない)
  - 第4正規形(多値従属性はない)
  - 第5正規形(結合従属性はない)

具体例(9)

- 初期スキーマ
  - 履修(学生番号、科目番号、氏名、所属学部、所属学科、住所、科目名、単位数、成績)
- 正規化後のスキーマ
  - 履修(学生番号、科目番号、成績)
  - 学生(学生番号、氏名、所属学科、住所)
  - 学科(所属学科、所属学部)
  - 科目(科目番号、科目名、単位数)

補足：第１正規形を満たさない場合の分解

- 初期スキーマ
  - 従業員（従業員番号, 氏名, 年齢,
    電話番号1, ・・・, 電話番号n）
    - 一人の従業員が複数（不定数）の電話番号の値を持つ
    - 第１正規形を満たさない
- 第１正規形を満たすように分解
  - 従業員（従業員番号, 氏名, 年齢）
  - 電話番号（従業員番号, 電話番号）
    - 全属性の組み合わせが候補キーとなる
    - 従業員番号の値は同じで、電話番号の値が異なるインスタンスを、複数格納できる

補足：第１正規形を満たさない場合の分解

- 初期スキーマ
  - 従業員（<u>従業員番号</u>, 氏名, 年齢,
    電話番号1, ･･･, 電話番号n）
    - 一人の従業員が複数（不定数）の電話番号の値を持つ

従業員番号	部門番号	氏名	年齢	電話番号
001	1	織田 信長	48	0948-29-1111
002	2	豊臣 秀吉	45	0948-29-2222 090-1234-3333
003	3	徳川 家康	39	0948-29-4444 090-1234-5555 080-5678-6666

第１正規形を満たすように分解

補足：第１正規形を満たさない場合の分解

- 第１正規形を満たすように分解
  - 従業員（従業員番号, 氏名, 年齢,
  - 電話番号（従業員番号, 電話番号）

従業員番号	部門番号	氏名	年齢
001	1	織田 信長	48
002	2	豊臣 秀吉	45
003	3	徳川 家康	39

従業員番号	電話番号
001	0948-29-1111
002	0948-29-2222
002	090-1234-3333
003	0948-29-4444
003	090-1234-5555
003	080-5678-6666

リレーションスキーマの設計のまとめ

- 概念設計の方法
  - 実体関連モデルを用いる方法
- 論理設計の方法
  - 実体関連モデルからスキーマを決定する方法
  - 仮のスキーマを正規化していく方法

論理設計の方法のまとめ

1. 実体関連モデルからスキーマを作る方法
2. スキーマを正規化していく方法

- 通常は、1の方法でスキーマを設計し、必要に応じて、2の正規化を行うのが一般的
  - きちんとした実体関連モデルからスキーマを作成していれば、正規化の必要はないことが多い
- それほど実用では使わないが、2の方法をきちんと身につけておくことは重要

レポート課題（前半）

レポート課題

- 自分で決めた何らかのテーマを題材にして、データベースとWebインターフェースを作成

1. データベースのスキーマの設計
   - 全ての正規形を満たすように正規化を行う
2. データベースの作成
   - テーブルの作成、データの追加
3. データベースへの問い合わせ
   - 問い合わせの具体例を考えてSQLを作成・実行

今回説明

4. Webインターフェースの作成
   - 一覧表示、追加・削除・更新、実用的な検索

後日説明

レポート課題の提出

- 2回に分けて提出する
- 第1回提出
  - 授業期間中の提出締切（詳細はMoodleを参照）
  - 課題1・2 の内容のみを提出（配点：35点）
- 第2回提出
  - 授業期間後の提出締切（詳細はMoodleを参照）
  - 課題3・4 の内容を追加して提出（配点：65点）
    - 課題1・2の変更も受け付ける（第1回分の点数に考慮）
  - 第1回の未提出者は、提出を受け付けない

レポート課題資料

- 演習レポート課題説明
- レポート LaTeX テンプレート ＋ サンプル（PDF）
  - テンプレートに従って作成する
  - LaTeX が使用できない場合は、指定されている通りの項目や形式が正しく守られていれば、別のソフトウェアを使って作成しても構わない
- 課題説明資料やテンプレートに明記されていない項目は、各自で判断して作成すること
  - 資料で説明されている以上の個別の作成方法や評価基準についての質問には、回答しない

レポート課題資料

- 演習レポート課題説明
- レポート LaTeX テンプレート ＋ サンプル（PDF）
  - テンプレートに従って作成する
  - LaTeX が使用できない場合は、指定されている通りの項目や、[illegible]が正しく守られていれば、別のソフトウェアを使って作成[illegible]構わない
- 課題[illegible]ていない項目は、各自で判断して作成すること
  - 資料で説明されている以上の個別の作成方法や評価基準についての質問には、回答しない

これらの資料は、レポート課題全体の内容を含む
後半の内容は、後日の講義で説明する

LaTeX 使用方法

- LaTeX の基本的な使用方法は、１年生の「プログラミング（リテラシー）」で学習した通り
- その他の使用方法は、レポート課題説明の付録の解説や、テンプレートの例を参照
  - コンパイル時のエラーが出た場合も、エラーへの対処方法の説明を参照する
- 図の挿入方法も、テンプレートを参照する
  - スクリーンショットは、GIMPなどのソフトウェアを使って、トリミング等の編集を行うことができる
  - 画像は、EPS形式で保存したものを、LaTeX ソースファイルから参照して、挿入することができる
  - 同じく、テキストやソースファイルも、ソースファイルに挿入できる
  - ラベルを使って図やソースファイルを参照すると、番号が自動更新される
- LaTeX は、今後もレポートや卒業論文作成等で使う機会があるので、慣れておくと良い

レポート課題

- 自分で決めた何らかのテーマを題材にして、データベースとWebインターフェースを作成

1. データベースのスキーマの設計
   - 全ての正規形を満たすように正規化を行う
2. データベースの作成
   - テーブルの作成、データの追加
3. データベースへの問い合わせ
   - 問い合わせの具体例を考えてSQLを作成・実行
4. Webインターフェースの作成
   - 一覧表示、追加・削除・更新、実用的な検索

1. スキーマの設計

- 思いつく全ての属性を挙げて1つの初期スキーマとし、候補キーや関数従属性を列挙
  - 格納されるデータの条件を書き出す
  - 条件にもとづいて、候補キーや関数従属性を書き出す
- 各正規形を満たすかどうかを順番に検証
  - ・・・より、第？正規形を満たす　or
    ・・・より、第？正規形を満たさないため、分解
- 最終的に得られたスキーマを示す

1. スキーマの設計

- 注意：必ず最初に1つの初期スキーマを挙げて、段階的に分解していくこと
  - 正規化の練習なので、最初から正規化済みの複数のスキーマを挙げているものは減点とする
- 注意：無理に複雑な正規化を行う必要はないので、最初は、第3正規形にもとづく1回の正規化を行うようなスキーマを設計する
  - 分解後のリレーションの数は、３つ以内（２ or ３個）になるようにすることを推奨する
    - リレーションの数が多くなると、後のWebインターフェースの作成で苦労するため

2. データベースの作成

- テーブルの作成
  - 設計したリレーションスキーマをもとに、複数のテーブルを作成する
  - テーブル名、属性名は、適切な英単語（アルファベット）に変更する
    - PostgreSQL の予約語は使えないので注意する
- データの追加
  - インターフェースのテストに必要な最低限のデータを追加する（最低20個程度）
- レポートには、テーブルの作成に使用したコマンド・SQLやデータ、格納結果を示す

2. データベースの作成

- 注意：データベースは、これまでの演習で作成したものを用いること
  - 自分のアカウント名（九工大ID）のデータベース
  - 勝手に新しいデータベースを作成しないこと
  - こままでに作成した従業員・部門テーブルは、そのままで構わない

3. データベースへの問い合わせ

- 自分が作成したデータベースに対して、問い合わせの例を考えて、SQLを記述し、実行する
  - 最低3つの問い合わせの例を示す
  - なるべく実用的な例（データベースを利用するときに使われることが想像できる例）を選ぶ
    - 問い合わせには、条件に使用する属性が異なる問い合わせや、結合を用いた問い合わせを含める
    - GROUP BY（+HAVING）や入れ子型問い合わせ（相関あり・なし）などの、授業で学習したやや複雑な構文を用いたSQLを含めることが望ましい

4. Webインターフェースの作成

- 後日の講義で説明
- その他の注意点等も、後日の講義で説明

評価基準

- レポート課題の説明に書かれている各課題ごとの評価項目を参照
  - 説明に書かれている以上の、各項目の詳細な判断基準や点数に関しての質問は受け付けないので、各自で判断すること

レポート課題（前半）の提出

- 2回に分けて提出する
- 第1回提出
  - Moodleの課題提出ページから提出
    - PDF形式のファイルを指定されたファイル名で提出
      - 他の提出方法は受け付けないので注意する
  - 授業期間中の提出締切（詳細はMoodleを参照）
  - 課題1・2 の内容のみを提出
    - テンプレートの課題3・4の部分は削除して提出する
    - 課題3・4 も終わっていれば含めても可（評価対象外）

まとめ

- 前回の復習
  - 正規形と正規化
    - 第2正規形～第５正規形
  - 正規形のまとめ
- 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
- レポート課題（前半）

全体のまとめ

- リレーションスキーマの正規形と正規化
  - 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１正規形～第５正規形
- 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

次回予告

- PHPによるWebインターフェース開発演習
- 次回（第10回）
  - WWWの仕組み
  - HTML＋PHP の基礎
  - 演習方法・手順
  - PHPによるインターフェース開発（1）
- 次々回（第11回）
  - PHPによるインターフェース開発（2）
  - レポート課題