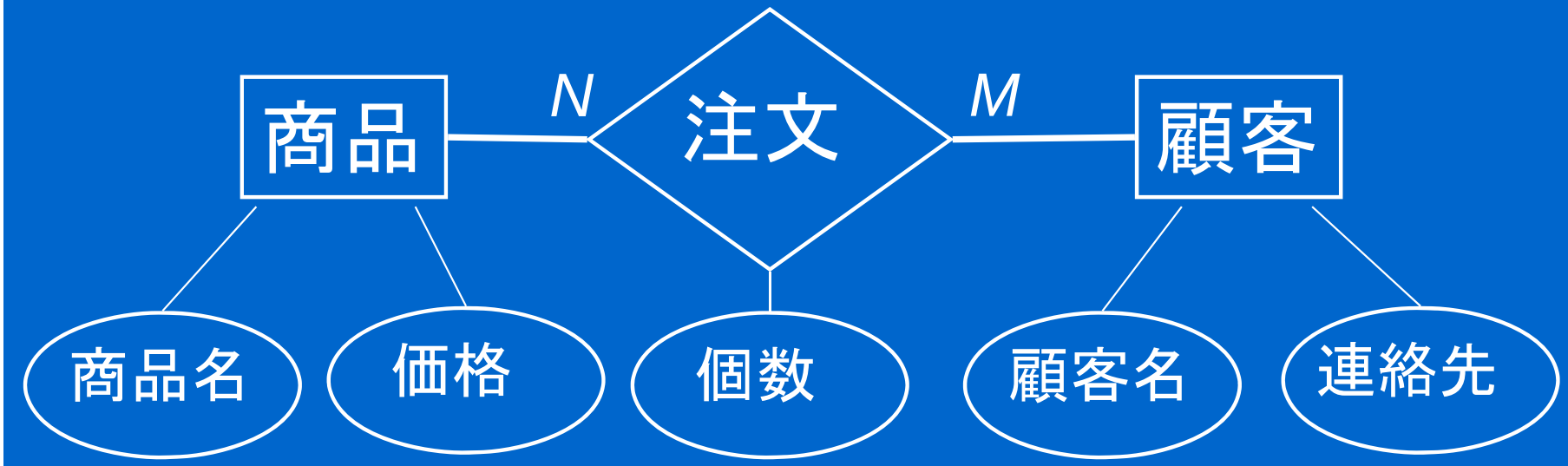
リレーションスキーマの設計 (Design of Relation Schema)

リレーションスキーマを設計するときの基本的な方針として、次の2種類がある。

1. 多数の属性を持つ一つ(または少数)の大きなリレーションにまとめて、
   検索時に**射影演算**により必要な属性を取り出す方法、
2. 少数の属性しかない多数の小さなリレーションに分割して、
   検索時に必要に応じて**結合演算**によりリレーションを結合することで必要な属性を組み合わせて行く方法

リレーショナルスキーマの設計の例

- 実体(entity):実世界に存在する事物に対応
- 関連(relationship):実体と実体の間にある関係
- 属性(attribute):実体や関連の持つ性質(名前や数など)



実体

関連(1対1、1対多、多対1、多対多などがある)

属性

設計方針1（1つのリレーション）の例

注文

商品名	価格	顧客名	連絡先	個数
冷蔵庫	1000	松田	4390	1
テレビ	500	松田	4390	3
テレビ	500	瀬尾	4391	2
エアコン	1200	瀬尾	4391	1

利点

- 問合せが射影演算と選択演算で表現でき、**結合演算の必要がない**
- 商品番号、顧客番号という外部キーによる参照が不要

欠点

- **更新（挿入、削除、修正）で異状が発生する可能性がある**（後述）

設計方針2（複数のリレーション）の例

商品

商品番号	商品名	価格
G1	冷蔵庫	1000
G2	テレビ	500
G3	エアコン	1200

顧客

顧客番号	顧客名	連絡先
C1	松田	4390
C2	瀬尾	4391

注文

注文番号	商品番号	顧客番号	個数
O1	G1	C1	1
O2	G2	C1	3
O3	G2	C2	2
O4	G3	C2	1

欠点
- 問合せで結合演算が必要になる
- 商品番号、顧客番号という**外部キーによる参照が必要**

利点　**更新時の異状が発生しない**（後述）

更新時異状(update anomalies)

- **更新時異状**とは、データベースの更新において必要とされるデータ操作が、一貫性制約に反するため行えない状況を指す
- 次の3種類がある

1. **タップル挿入時異状(insertion anomaly)**：必要なタップルの挿入が一貫性制約に反するためできない
2. **タップル削除時異状(deletion anomaly):** 必要なタップルの削除が一貫性制約に反するためできない
3. **タップル修正時異状(modification anomaly):** 必要なタップルの修正が一貫性制約に反するためできない

更新時異状の例

リレーション 受講

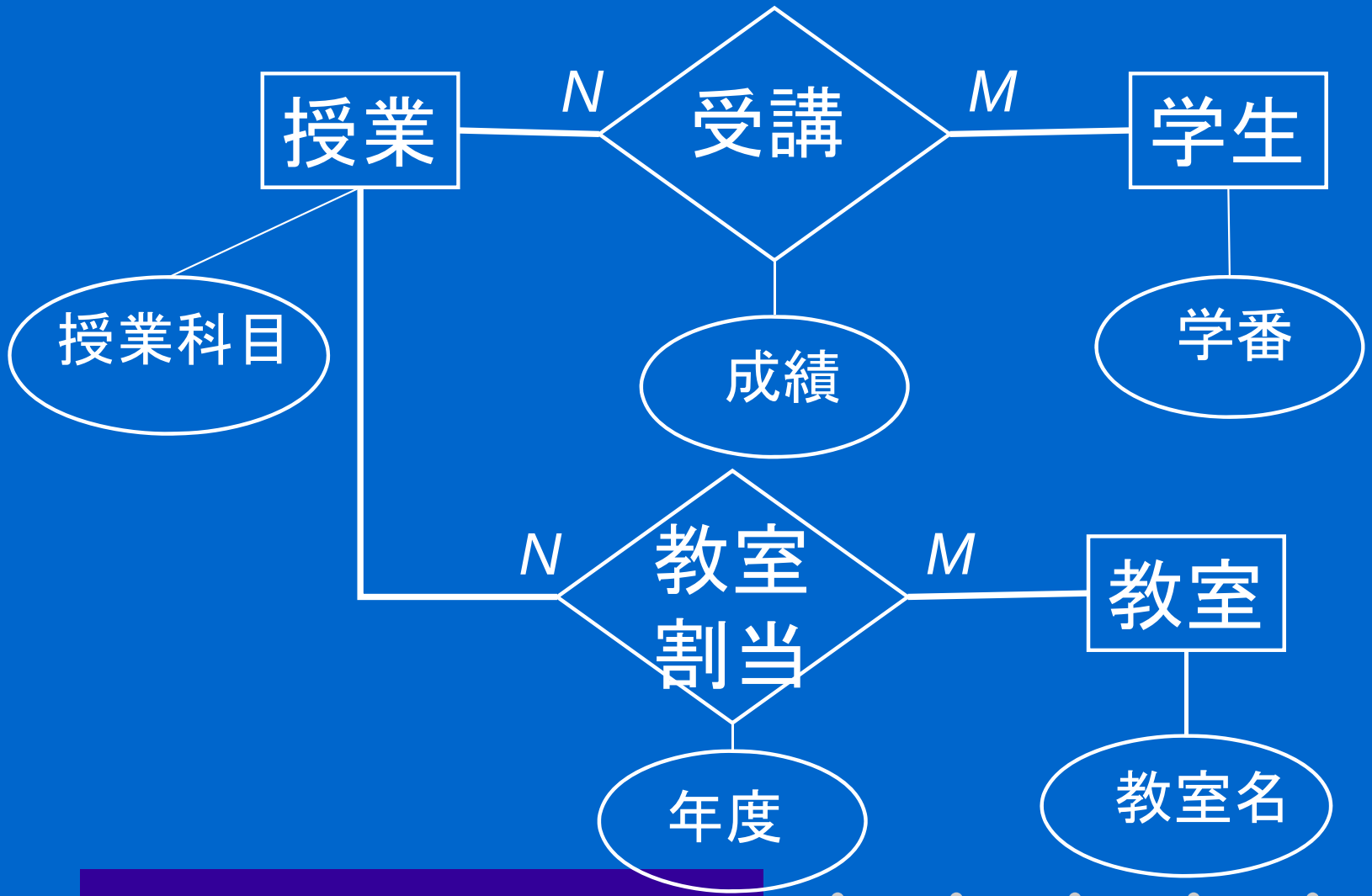
関数従属性：{授業科目、学番} →成績、授業科目→教室

{授業科目、学番}が主キー（唯一の候補キー）

授業科目	学番	成績	教室
データベース	S01	85	401
データベース	S02	90	401
演習	S03	NULL	演習室

更新時異状の例の概念モデル

- 複数の関連を1つにまとめている



タップル挿入時異状の例

受講 （関数従属性：{授業科目,学番} ➔成績, 授業科目➔教室）
{授業科目, 学番}が主キー

授業科目	学番	成績	教室
データベース	S01	85	401
データベース	S02	90	401
演習	S03	NULL	演習室
実験	NULL	NULL	実験室

- **タップル挿入時異状**： 教室は授業科目だけで決まるので、まだ受講がない教室の情報（例：実験、実験室）を登録しようとして、（実験,NULL,NULL,実験室）を挿入すると、**主キー制約に違反**するのでできない

タップル削除時異状の例

受講 （関数従属性：{授業科目,学番} ➔成績, 授業科目➔教室）
{授業科目, 学番}が主キー

授業科目	学番	成績	教室
データベース	S01	85	401
データベース	S02	90	401
~~演習~~	~~S03~~ →NULL	~~NULL~~	~~演習室~~

×

- **タップル削除時異状**： S03が受講を取り止めたためS03を含むタップルを消そうとして（演習、S03、NULL、演習室）を削除すると、演習を演習室で実施することまで消えてしまう（S03を空値にしたいが**主キー制約に違反**するためできない）

タップル修正時異状の例

受講 （関数従属性：{授業科目,学番} ➔成績, **授業科目➔教室**）
{授業科目, 学番}が主キー

授業科目	学番	成績	教室
データベース	S01	85	401➔201
データベース	S02	90	401➔201
演習	S03	NULL	演習室

- **タップル修正時異状**： データベースの教室を201に変更しようとすると、一度に複数のタップルの修正が必要になる

なぜ更新時異状が起きるか？

- 関数従属性は、一貫性制約の一つであり、常に成立する
- 関数従属性を決定している属性集合（関数従属性X→Yの場合はX）が、
  - 候補キーKと同一（X=K）になっているか、
  - 超キーになっている（候補キーを含んでいる、K⊆X）

  ときは、更新時異状は起きない
- 左辺が候補キーまたは超キーになっていない関数従属性があると、そこで更新時異状が起きる

情報無損失分解(information lossless decomposition)

- リレーションスキーマ**R**(X,Y,Z,V)で、{X Y}→Z, X→Vという関数従属性が成り立つ場合には更新時異状が起こる
- **R**のリレーション*R*を、$R1 = \pi_{XYZ}(R)$と$R2 = \pi_{XV}(R)$に分解した後、更新すれば更新時異状は発生しない
- この分解をしても、*R*の属性X, Y, ZのデータはR_1から検索できるし、R_1にない属性Vのデータについても、元の*R*において関数従属性X →Vが成り立っていることから $R_1 \bowtie R_2$（R_1とR_2の自然結合）により参照できる
- このように自然結合により元のリレーションを復元できるような分解を情報無損失分解と呼ぶ。

多値従属性に関する情報無損失分解の条件

Rをリレーション、XとYを$X \cup Y = \Omega_R$かつ$X \cap Y \neq \phi$であるような属性集合とするとき、Rが射影$\pi_X(R)$と$\pi_Y(R)$に情報無損失分解されるための必要十分条件は、

Rで多値従属性$X \cap Y \twoheadrightarrow X|Y$が成り立つことである

関数従属性に関する情報無損失分解の条件

Rをリレーション、XとYを$X \cup Y = \Omega_R$かつ$X \cap Y \neq \phi$であるような属性集合とするとき、Rが射影$\pi_X(R)$と$\pi_Y(R)$に情報無損失分解されるための十分条件は、

Rで関数従属性$X \cap Y \rightarrow X - Y$、または

$X \cap Y \rightarrow Y - X$が成り立つことである

情報無損失分解の判定例

関数従属性：｛授業科目、学番｝→成績、
授業科目→教室

分解：X=｛授業科目、学番、成績｝、
Y=｛授業科目、教室｝は、

X∩Y={授業科目}、Y-X={教室}で、

X∩Y→Y－Xが成り立つので、情報無損失分解である

情報無損失分解の意味

関数従属性：｛授業科目、学番｝→成績、授業科目→教室
分解：X=｛授業科目、学番、成績｝、Y=｛授業科目、教室｝
は、X∩Y={授業科目}、Y-X={教室}とすると、
分解しても、X∩Yで参照できているので、自然結合で元に戻せる

授業科目	学番	成績
データベース	S01	85
データベース	S02	90
演習	S03	NULL

授業科目	教室
データベース	401
演習	演習室

情報無損失分解による更新時異状の解消

関数従属性：{授業科目、学番}→成績、　授業科目→教室　に応じて分解
　　{授業科目、学番}が主キー　　授業科目が主キー

授業科目	学番	成績
データベース	S01	85
データベース	S02	90
~~演習~~	~~S03~~	~~NULL~~

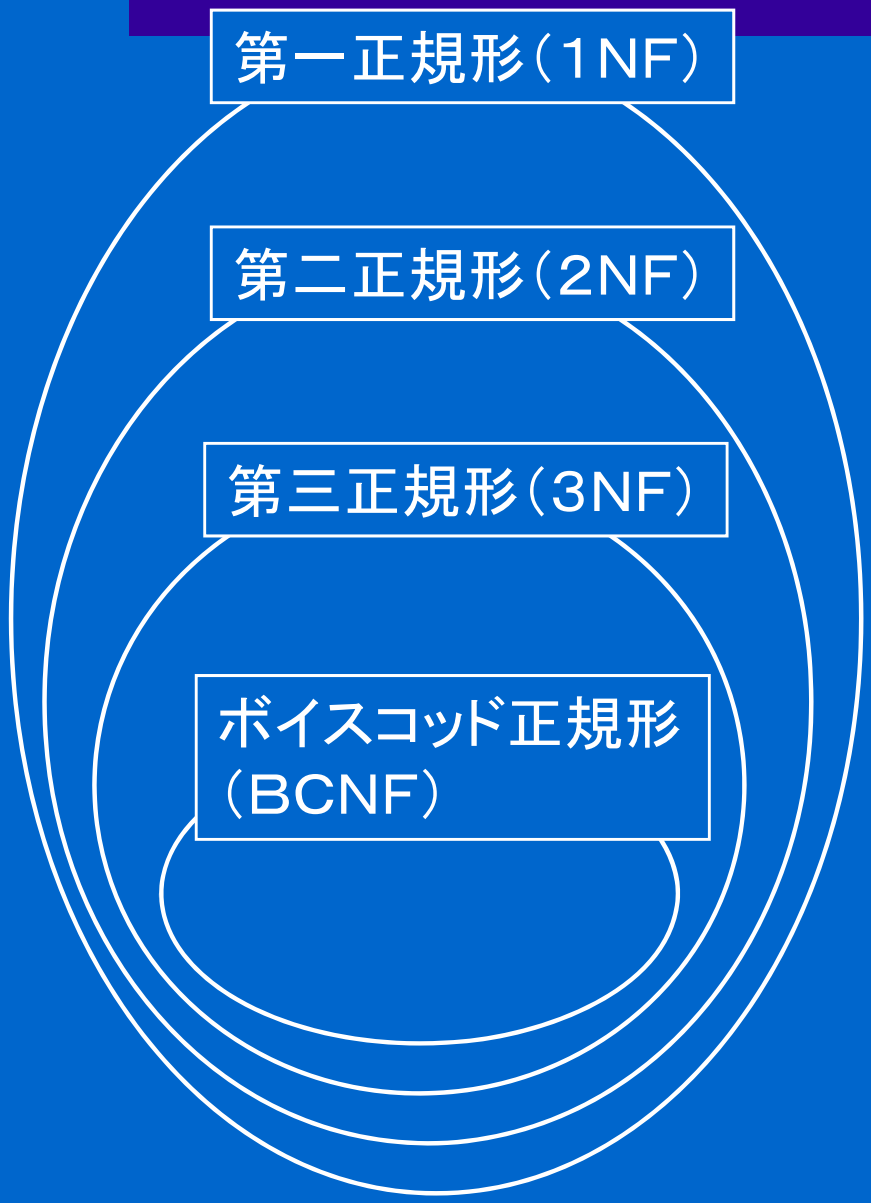
授業科目	教室
データベース	401→**201**
演習	演習室
実験	**実験室**

- **タップル挿入時**：　教室の情報（実験、実験室）のみを登録
- **タップル削除時**：　（演習、S03、NULL）を削除しても教室が演習室である情報は消えない
- **タップル修正時**：　データベースの教室の変更はそこだけ修正
- 分解後のリレーションを**自然結合で結合すると分解前のリレーションと同じ**になる：分解により情報が欠損しないので、**情報無損失分解**

リレーショナルデータベースの正規化理論

- あるリレーションで更新時異状が起こるとき、情報無損失分解を行えばその異状が生じないようにすることができる場合がある
- リレーションが第一正規形であるというだけでは、更新時異状が起こる恐れがあるため、さらなる分解（**高次の正規形**）が必要

関数従属性についての高次の正規形



すべての**非キー属性**がキーに**完全**関数従属か？

N → 第一正規形（1NF）

Y →

すべての**非キー属性**についてキーへの**推移的**な関数従属性が**ない**か？

N → 第二正規形（2NF）

Y →

すべての関数従属性は**自明**かまたは左辺が**超キー**か？

N → 第三正規形（3NF）

Y → ボイスコッド正規形（BCNF）

非キー属性とは？

- リレーションの属性のうちで、いずれの「候補キー」にも含まれないものを**非キー属性**とよぶ

（参考）キーのまとめ

- **超キー (super key)**：次の条件を満たす属性集合K

  条件1 $(\forall\ t, t' \in R)(t[K]=t'[K] \Rightarrow t=t')$

  - 超キーによりリレーションの全属性集合の値が決まる（＝タップルが一意に定まる）
- **候補キー (candidate key)**：超キーのうち最小のもの（属性集合を一つでも除くと条件1が満たされないもの）
- **主キー (primary key)**：あるリレーションの候補キーのうち、空値を取らず、主要なもの（どれが主要かはデータベースの設計者が決める）
- **外部キー (foreign key)**：他のリレーションで、主キーとなっているもの

第二正規形 (second normal form, 2NF)

リレーションスキーマRが**第二正規形**であるとは、次の二つの条件を満たすときをいう

1. Rは第一正規形である
2. Rのすべての非キー属性は、Rの各候補キーに完全関数従属している

(参考)完全関数従属性 (full functional dependency)

- リレーションスキーマRの属性集合X, Y (XとYは互いに素である必要はない)について、次の2つの条件がともに成立するとき、「**Xからの完全関数従属性**が存在する」(または「**YはXに完全関数従属している**」)という。

1. X → Y,
2. Xのいかなる真部分集合X'に対してもX' → Yが成立しない.

第二正規形かどうかの判定例(1)

与えられた条件

リレーションスキーマ 受講(授業科目、学番、成績、教室)
関数従属性：{授業科目、学番} →成績、
授業科目 →教室

候補キーは{授業科目、学番}(主キーでもある)
非キー属性は、成績 と 教室
成績 は候補キーに完全関数従属しているか？
教室 はどうか？

授業科目	学番	成績	教室
データベース	S01	85	201
データベース	S02	90	201
演習	S03	NULL	演習室

第二正規形かどうかの判定例(2)
(第一正規形だが第二正規形でない例)

リレーションスキーマ受講(授業科目、学番、成績、教室)
関数従属性:{授業科目、学番}→成績、
授業科目→教室

候補キーは{授業科目、学番}(主キーでもある)
非キー属性は、成績 と 教室
○成績 は候補キーに完全関数従属している
×教室 は候補キーに(関数従属はしているが)**完全関数従属ではない**

授業科目	学番	成績	教室
データベース	S01	85	201
データベース	S02	90	201
演習	S03	NULL	演習室

情報無損失分解

関数従属性：{授業科目、学番}→成績、 授業科目→教室　に応じて分解

R1:{授業科目、学番}が主キー

授業科目	学番	成績
データベース	S01	85
データベース	S02	90
演習	S03	NULL

R2:授業科目が主キー

授業科目	教室
データベース	201
演習	演習室

X={授業科目、学番、成績}、Y={授業科目、教室}とおくと、
X∩Y={授業科目}、
Y-X={教室}で、X∩Y→Y−Xが成り立つので、情報無損失分解である
R1の非キー属性 成績 も、R2の非キー属性 教室 も、それぞれのリレーションスキーマの候補キー（主キー）である{授業科目、学番}, 授業科目に、それぞれ完全関数従属しているので、少なくとも**第二正規形**
さらに高次の正規形になっている可能性があることに注意

第三正規形 (third normal form, 3NF)

リレーションスキーマRが第三正規形であるとは、次の2つの条件を満たすときをいう

1. Rは第二正規形である
2. Rのすべての非キー属性は、Rのいかなる候補キーにも推移的に関数従属しない

（参考）推移的関数従属性 (transitive functional dependency)

リレーションスキーマ**R**の属性集合X, Yについて次の3つの条件が成立するとする

1. X → Y
2. Y ↛ X (Y → Xが成り立っていないことを表す)
3. Y → Z

すると、次が成立する

- X → Z
- Z ↛ X

このとき，新しく得られた X → Z という関数従属性を**推移的関数従属性**という

第三正規形かどうかの判定例(1)

与えられた条件

リレーションスキーマ配属(社員番号，プロジェクト名，所属グループ)

関数従属性: 社員番号→プロジェクト名，
プロジェクト名→所属グループ

- 候補キーは、社員番号 となる（主キーにもなっている）
- 非キー属性は、プロジェクト名と所属グループ

このリレーションスキーマは、第二正規形か？

第三正規形か？

：第三正規形かどうかの判定例(2)

（第二正規形だが第三正規形でない例）

リレーションスキーマ配属(社員番号，プロジェクト名，所属グループ)
関数従属性: 社員番号→プロジェクト名，
プロジェクト名→所属グループ

- 非キー属性 プロジェクト名、所属グループは、どちらも候補キー社員番号に完全関数従属している（第二正規形 ○）
- 非キー属性「所属グループ」は、候補キー「社員番号」に推移的に従属している（第三正規形×）
- プロジェクト名→所属グループ で、更新時異状が起こる
- $\pi_{\text{社員番号，プロジェクト名}}$(配属)と$\pi_{\text{プロジェクト名，所属グループ}}$(配属)への情報無損失分解で、推移的な関数従属性がなくなり、第三正規形に分解（実際には、ボイスコッド正規形にまで分解）

ボイスコッド正規形 (Boyce-Codd normal form, BCNF)

リレーションスキーマRが **ボイスコッド正規形** であるとは、次の条件が成立するときをいう

X→YをRの任意の関数従属性とするとき、

- X→Yは自明な関数従属性であるか、または、
- XはRの超キーである（XはRの候補キーを含んでいる）

が成り立つ

ボイスコッド正規形かどうかの判定例

与えられた条件

リレーションスキーマ受講(学生名，科目名，教員名)
関数従属性: {学生名，科目名}→教員名，
教員名→科目名.

- 候補キーは、{学生名，科目名}と {学生名，教員名}

なぜ、{学生名，教員名}が候補キーとなるのか？

候補キーの求め方(1)

リレーションスキーマ**R**(A1, A2, ..., An)

1. 自明でない関数従属性なし
   - 全属性集合Ω_R={A1, A2, ..., An}のみが唯一の候補キー（非キー属性は**なし**）
2. 自明でない関数従属性あり
   - 2.1 自明でない関数従属性X→Y（X, Yは属性集合）が1個しかない
     ⇒ Ω_Rは候補キーではなく、属性集合K= Ω_R -(Y-X)が候補キー（**Y-X**の要素が非キー属性）
   - 2.2 自明でない関数従属性が2個以上ある
     ⇒ Ω_Rは候補キーではない
     候補キーとなる属性集合が複数個ある可能性がある

（参考）自明な関数従属性 (trivial functional dependency)

- 次の関数従属性は常に成立するため、自明な関数従属性と呼ばれる。

1. Y ⊆ Xのとき、X → Y,
2. X → ϕ ,
3. X → X.

候補キーの求め方(2)

リレーションスキーマ**R**(A1, A2, ..., An)

2.2 自明でない関数従属性が**2**個以上ある

X1→Y1, X2→Y2, ..., Xm→Ym

2.2.1 X1∪X2∪・・・∪Xm と Y1∪Y2∪・・・∪Ymが互いに素

⇒候補キーは**Ω**R- (Y1∪Y2∪・・・∪Ym)

Y1∪Y2∪・・・∪Ymの要素が非キー属性

2.2.2 X1∪X2∪・・・∪Xm と Y1∪Y2∪・・・∪Ymが互いに素**ではない**

2.2.2.1 任意のXiとYjで Xi∩Yi = Xj ∩ Yj = φ であり、包含関係は **Yi ⊇ Xj のみ**である(Xi⊇ Yj はない)

Xi→Yi, Xj→Yj で推移律が成立

2.2.2.2 2.2.2.1の条件以外の包含関係がある

候補キーの求め方(3)

リレーションスキーマ**R**(A1, A2, ..., An)

2.2.2.1 任意のXiとYjでXi∩Yi = Xj∩Yj = φ で、包含関係はYi⊇Xj **のみ**である(**Xi⊇Yjはない**)

(Xi→Yi, **Yi→Xj**, Xj→Yj で推移律が成り立つ)

⇒候補キーは**ΩR-** (Y1∪Y2∪・・・∪Ym)

Y1∪Y2∪・・・∪Ymの要素が非キー属性

2.2.2.2任意のXiとYjとの包含関係は、2.2.2.1の条件以外にある

⇒**ΩR-** Yi、**ΩR-** Yj、**ΩR-** (Yi∪Yk), ・・・など候補キーが多数出てくる可能性がある

どのYiの要素も非キー属性とならないことがある(Xi ⊇ Yj となって候補キーに含まれる可能性がある)

ボイスコッド正規形の判定例(1)

与えられた条件

リレーションスキーマ受講(学生名，科目名，教員名)
関数従属性: {学生名，科目名}→教員名，
教員名→科目名.

- Xi⊇Yj (i≠j)がある（Xi={学生名，科目名}，Yj=科目名）
  2.2.2.2の条件にあたり候補キーが多数出てくる
- 候補キーは、{学生名，科目名}と {学生名，教員名}
- 非キー属性はなし

このリレーションスキーマは、第二正規形か？第三正規形か？ボイスコッド正規形か？

ボイスコッド正規形の判定例(2)
（第三正規形であってボイスコッド正規形でない例）

リレーションスキーマ受講(学生名，科目名，教員名)
関数従属性: {学生名，科目名}→教員名，
教員名→科目名.

- 非キー属性がないときは、**第三正規形となる（第二正規形でもある）**
- 関数従属性 教員名→科目名 は自明な関数従属性ではなく、かつ 教員名 は超キーではないので、**ボイスコッド正規形ではない**

ボイスコッド正規形の判定例(3)

（第三正規形であってボイスコッド正規形でない例）

情報無損失分解

$\pi_{教員名,\ 科目名}$(受講)、

$\pi_{教員名,\ 学生名}$(受講)に分解

分解により、ボイスコッド正規形となる

（注意）この分解は、もとのリレーションスキーマが持っていた関数従属性 {学生名，科目名} → 教員名 が保存されない

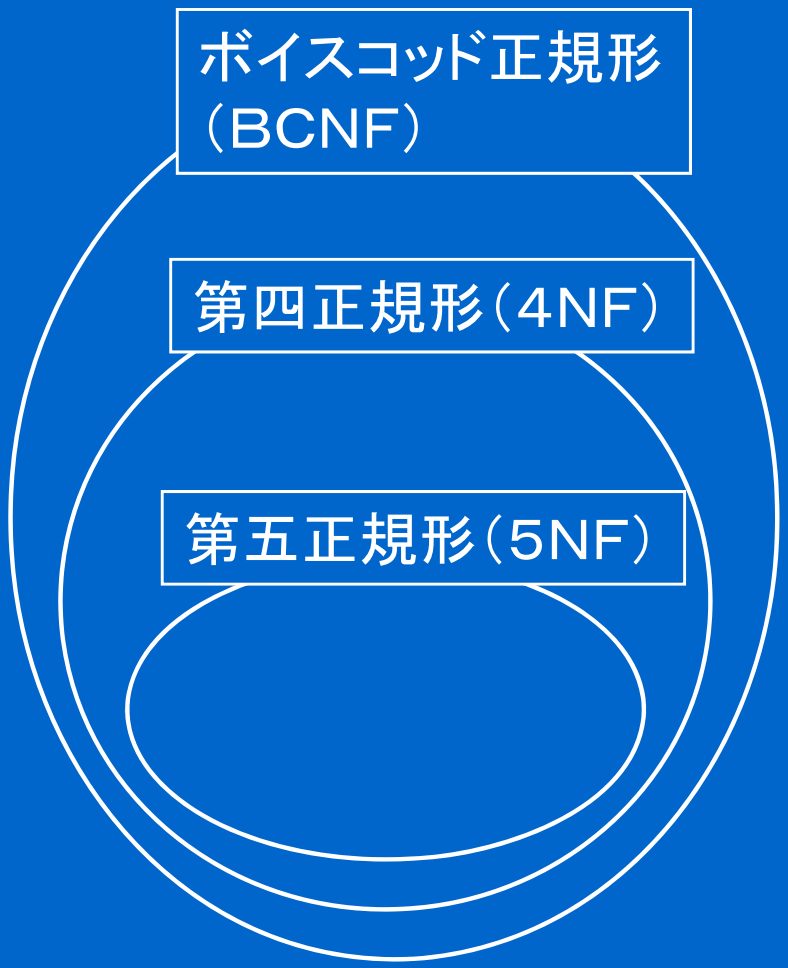
第三正規形までへの分解では関数従属性が保存されるが、それ以上高次の正規形への分解では、関数従属性が保存されないことがある

：それぞれどの正規形になるか？

リレーションスキーマ**R**(A1, A2, …, An)

1. 自明でない関数従属性なし
   - ➢ 全属性集合Ω_R={A1, A2, …, An}のみが唯一の候補キー
     ボイスコッド正規形*
2. 自明でない関数従属性あり
   - 2.1（自明でない）関数従属性X→Y（X, Yは属性集合）が1個だけ成り立っている
     - ⇒ Ω_Rは候補キーではなく、属性集合K= Ω_R -(Y-X)が候補キー
       K=Xならボイスコッド正規形*、**そうでなければ**第一正規形
   - 2.2（自明でない）関数従属性が2個以上成り立っている
     - 2.2.1 XとYは互いに素　第一正規形か**ボイスコッド正規形***
     - 2.2.2 互いに素ではない
       - 2.2.2.1 推移的な関数従属性のみ　**第二正規形**
       - 2.2.2.2 推移的な関数従属性以外にある　**正規形はスキーマに依存**

【参考】関数従属性以外の高次の正規形



すべての関数従属性は自明
かまたは左辺が超キーか？

N → 第三正規形（3NF）

Y ↓

すべての**多値従属性**は自明
かまたは左辺が超キーか？

N → ボイスコッド正規形（BCNF）

Y ↓

すべての**結合従属性**は自明
かまたは各項が超キーか？

N → 第四正規形（4NF）

Y ↓

第五正規形（5NF）

【参考】第四正規形 (forth normal form, 4NF)

リレーションスキーマRが**第四正規形**であるとは、次の条件が成立するときをいう。

X →→ YをRの任意の多値従属性とするとき、

- X →→ Yは自明な多値従属性であるか、または、
- XはRの超キーである（XはRの候補キーを含んでいる）

（多値従属性 →→ を関数従属性→に変更すれば、ボイスコッド正規形の定義になる）

【参考】第五正規形(fifth normal form, 5NF)

リレーションスキーマ**R**が**第五正規形**であるとは、次の条件が成立するときをいう

＊(X_1, X_2, …, X_n) を**R**の任意の結合従属性とするとき、

- ＊(X_1, X_2, …, X_n)は自明な結合従属性であるか、または、
- 各X_i ($1 \leqq i \leqq n$)は**R**の超キー（候補キーかまたは候補キーを含む）である

結合従属性(join dependency, JD)

- $X_1, X_2, \ldots, X_n$を，$X_1 \cup X_2 \cup \ldots \cup X_n = \Omega_R$となるような、リレーションスキーマ**R**の任意の属性集合とすると、**R**の任意のインスタンスRで、

  $R = \pi_{X1}(R) \bowtie \pi_{X2}(R) \bowtie \ldots \bowtie \pi_{Xn}(R)$

  が成り立つとき、**R**で結合従属性が成立するといい，＊(X1, X2, …, Xn) と表す