データベース 講義資料　第7回 リレーションスキーマの設計（1）

九州工業大学 情報工学部　講義担当：尾下真樹

1. リレーションスキーマの設計

データベースを利用するためには、データベースにどのようなデータを格納したいかにもとづいて、リレーションスキーマの設計を行う必要がある。具体的には、どのようなリレーションを定義し、それぞれのリレーションにどのような属性や制約を持たせるか、ということを決める必要がある。このとき、データの更新時に不整合が生じることのないようなリレーションスキーマを設計する必要がある。

2. 正規化の必要性

リレーションスキーマには、第1正規形という全てのリレーションスキーマが必ず満たさなければならない正規形に加えて、第2正規形～第5正規形の**正規形**がある。正規形を満たしていないリレーションスキーマは、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、異常）が生じる場合がある。

例えば、以下のようなリレーションがあるとする。（下線が引かれている属性は主キーを表す。）

営業（商品番号, 顧客, 社員, 販売価格）

ただし、各顧客について、1社の顧客を担当する社員は1人だけ、というルールがあるものとする。

このリレーションスキーマは、ある正規形の条件を満たしていないため、データを更新（修正、挿入、削除）しようとした時に、例えば以下のような問題（データの不整合）が生じる。

- ある顧客の担当社員が替わると、その顧客に関する全ての営業データの社員番号を変更する必要がある。（**修正不整合**）
- ある顧客の担当社員が決まったとしても、その顧客との具体的な取引商品がなければ、その情報を挿入できない。（主キーに含まれる商品番号の属性が空値のデータを挿入することは許されないため。）（**挿入不整合**）
- ある顧客の取引商品が一時的に全てなくなった時（その顧客に関する営業のデータが全てなくなった時）、その顧客をどの社員が担当しているか、という情報も一緒に失われてしまう。（**削除不整合**）

このように、リレーションスキーマが正規形を満たしておらず、更新不整合が生じる場合は、正規形のルールを満たすように**正規化**を行う必要がある。具体的には、1つのスキーマを適切な複数のスキーマに**分解**することで、正規化を行うことができる。このとき、正しい分解を行うと、分解後のリレーション同士を結合することで、もとのリレーションを得ることができる（**情報無損失分解**）。逆に、間違った分解を行ってしまうと、分解後のリレーション同士を結合しても、余計なデータが出現してしまい、もとのリレーションを得ることができない（**情報損失分解**）。

例えば、上記のリレーションの場合は、以下のような2つのリレーションに分解すると、正しい分解（情報無損失分解）となる。

販売（商品番号, 顧客, 販売価格）　顧客担当（顧客, 社員）

正規形のルールや、正規形を満たさない場合の正しい分解の仕方については、次回の講義で勉強する。

3. 関数従属性と多値従属性

正しい分解（情報無損失分解）を行うための基準となるのが、関数従属性や多値従属性などの属性間の制約である。

3.1. 関数従属性

あるリレーションにおいて、ある属性（または複数の属性の集合）Xと別の属性（または複数の属性の集合）Yとの間に、Xの値が決まればYの値が一意に決まるという制約があるとき（リレーション中にXの値が等しい複数のデータがあったとすると、その全てのデータのYの値も等しくなるとき）、YはXに関数従属し（XとYの間には**関数従属性**があり）、$X \rightarrow Y$ と表す。
例えば、上記のリレーションには、**顧客番号 → 社員番号** という関数従属性があることになる。

3.2. 多値従属性

あるリレーションにおいて、ある属性（または複数の属性の集合）Xと別の属性（または複数の属性の集合）Yとの間に、あるXの値に対応するYの値が複数あり、そのXの値について、Yの値とX,Y以外の属性の値の全ての組み合わせのデータがリレーション内に存在するという制約があるとき、YはXに関数従属し（XとYの間には**多値従属性**があり）、$X \rightarrow\rightarrow Y$ と表す。

例えば、以下のようなリレーションがあるとする。（下線が引かれている属性は主キーを表す。）

プロジェクト（プロジェクト, 社員, ミーティング日）

ただし、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。

このとき、例えば、プロジェクト p1 に、社員 e1,e2 が属しており、ミーティング日が月曜日,木曜日の 2 つであるとすると、このリレーションには（p1, e1, 月曜日）,（p1, e2, 月曜日）,（p1, e1, 木曜日）,（p1, e2, 木曜日）の 4 つのデータ（全ての所属社員とミーティング日同士を組み合わせたデータ）が存在することになる。
そのため、このリレーションには、**プロジェクト →→ 社員** の多値従属性がある。
（プロジェクト →→ ミーティング日、または、プロジェクト →→ 社員｜ミーティング日 とも書ける。）

多値従属性があることが、正しい分解（情報無損失分解）を行えることの、必要十分条件となる。関数従属性は、多値従属性の特殊なケース（多値従属性の Y の値が 1 つしかないもの）であると考えられるので、関数従属性も、正しい分解（情報無損失分解）を行えることの、十分条件となる。
なお、あるリレーションにどのような関数従属性・多値従属性が存在するかは、リレーションスキーマだけからは判断できないため、リレーションにどのようなデータが格納されるか（どのようなルールが存在するか）を考慮して、判断する必要がある。

3.3. 関数従属性の理論

あるリレーションに対して、一部の関数従属性が与えられているとき、以下のようなアームストロングの公理系を用いて、他の関数従属性を導出することができる。

- 反射律：Y が X の部分集合であれば X → Y　例：学生番号、氏名 → 氏名
- 増加律：X → Y であれば X∪A → Y∪A　例：学生番号 → 学科 なら、学生番号、氏名 → 学科、氏名
- 推移律：X→Y かつ Y→Z なら X→Z　例：学生番号 → 学科番号 かつ 学科番号 → 学科名 なら、学生番号 → 学科名

このうち、推移律が、自明ではない関数従属性を導出するという点で、特に重要である。