



3-3-2 データベース設計

p.211~

発表者:野口耕二郎

設計のアプローチ

- トップダウンアプローチ

- 全体から部分に落としこむアプローチ. 新しい要件や機能追加の場合に使用する
- 1. エンティティを洗い出し, 関係を考えながら E-R 図を作成する
- 2. エンティティの属性を洗い出し, 主キーを決定する
- 3. 正規化を行い, 多対多の関係を排除する

- ボトムダウンアプローチ

- 部分から全体にまとめていくアプローチ. 元のシステムが存在する場合に使用する
- 1. 帳票や仕様から属性の洗い出しを行う
- 2. 主キーを見つけ, テーブルの正規化を行う
- 3. テーブル構造から E-R 図を作成する

正規化

- 正規化とは, 正しい規則に従ってテーブルを分割すること
- 正規化の目的は更新時以上の排除
- テーブルを分けてデータの重複を排除し, 1 つのデータを 1 箇所に保管することでデータの整合性を保つ

問題

問題

データの正規化に関する記述のうち、適切なものはどれか。

- ア 正規化は、データベースへのアクセス効率を向上させるために行う。
- イ 正規化を行うと、複数の項目で構成される属性は、単一の項目をもつ属性に分解される。
- ウ 正規化を完全に行うと、同一の属性を複数の表で重複してもつことはなくなる。
- エ 非正規形の表に対しては、選択、射影などの関係演算は実行不可能である。

(平成22年春 応用情報技術者試験 午前 問31)

答え

解説

正規化を行う際、非正規形から第一正規形にすると、すべての属性がシンプル(単一の項目をもつ属性)になります。したがって、イが正解です。

ア アクセス効率は、正規化することによって低下することもあります。

ウ 同一の属性については、表同士の関係を表すために複数の表でもちます。

エ 関係演算は正規化の有無とは関係ありません。

《解答》イ

主キー・候補キー

- 候補キー: 全ての属性を一意に特定する属性または属性の組で最小のもの
 - (例) 社員(社員番号, 氏名, メールアドレス)
 - {氏名}は同姓同名というケースがあるため不適
 - 一意に決まりそうな{社員番号}と{メールアドレス}が候補キーとなる
- 主キー: テーブルの行を一意に特定するために候補キーの中から適当なものを選んでデータベースを実装する際に設定する
 - (例) 候補キーとして{社員番号}と{メールアドレス}があるが, {メールアドレス}が空値になる可能性があるため{社員番号}のほうに適している.

問題

問題

関係データベースの候補キーとなる列又は列の組に関する記述として、適切なものはどれか。

- ア 値を空値 (null) にすることはできない。
- イ 検索の高速化のために、属性の値と対応するデータの格納位置を記録する。
- ウ 異なる表の列と関連付けられている。
- エ 表の行を唯一に識別できる。

(平成22年秋 応用情報技術者試験 午前 問29)

答え

解説

候補キーは、表の行を唯一に識別できる列又は列の組なので、エが正解です。アの条件が加わると主キーになります。イはインデックスの説明、ウは外部キーの説明です。

《解答》エ

関数従属性

- 関数従属性: ある属性 A の値が決まったら別の属性 B の値も一意に決定できるということ.
- $A \rightarrow B$ と表記する
- (例) 商品番号 $\rightarrow$ 商品名という関数従属性があるとき, 商品番号“1001”が決まれば商品名“冷やし中華”も決まる

問題

問題

六つのタプルから成る関係Rの単一の属性間において成立する全ての関数従属性を挙げたものはどれか。ここで、 $X \rightarrow Y$ は、 X が Y を関数的に決定することを表す。

R

A	B	C
300	阿部商店	3
300	阿部商店	3
400	鈴木商店	2
400	鈴木商店	2
500	鈴木商店	1
500	鈴木商店	1

- ア $A \rightarrow B$
- イ $A \rightarrow C, C \rightarrow A$
- ウ $A \rightarrow B, A \rightarrow C, C \rightarrow A, C \rightarrow B$
- エ $A \rightarrow B, A \rightarrow C, B \rightarrow C, C \rightarrow A, C \rightarrow B$

答え

解説

関係(表) R を見ると、 A が300のときには B は必ず阿部商店、 A が400、500のときには B は必ず鈴木商店です。したがって、 A が決まれば B は一意に決まるので、 $A \rightarrow B$ の関数従属性は成り立ちます。

同様に、 A が300のときには C は必ず3、 A が400のときには C は必ず2、 A が500のときには C は必ず1です。したがって、 $A \rightarrow C$ の関数従属性は成り立ちます。

また A と C の場合は、逆から考えても、 C が3のときには A は必ず300など、1対1に対応しているので、 $C \rightarrow A$ の関数従属性も成り立ちます。

B と C の場合には、 C が3のときには B は阿部商店、 C が2と1のときには B は鈴木商店と、 C が決まれば B は一意に決定できます。つまり、 $C \rightarrow B$ は成り立ちます。

したがって、これらの四つの関数従属性を挙げているウが正解です。

正規化の手順



正規化の流れ

非正規形

- 1つのレコードのあるフィールドに複数の値が入れたテーブル

商品番号	商品	産地
100	りんご	青森県 長野県
101	みかん	和歌山県 愛媛県

第 1 正規形

- 1 つのレコードの全てのフィールドに 1 つだけ値が入っているテーブル

商品番号	商品	産地
100	りんご	青森県
100	りんご	長野県
101	みかん	和歌山県
101	みかん	愛媛県

第 2 正規形

- テーブルが第 1 正規形であり, 全ての非キー属性が各候補キーに完全関数従属している状態
- 完全関数従属性: $X \rightarrow Y$ で X の任意の部分集合 X_1 について $X_1 \rightarrow Y$ は成立しないときの従属性
- テーブルが完全関数従属の状態になるように情報無損失分解 (自然結合で元のテーブルに戻るようなテーブルの分解) を行う

情報無損失分解

候補キー以外に対する関数従属性 f_2 : 商品名 $\rightarrow$ 単価 を使って情報無損失分解

注文

顧客名	商品名	数量	単価	金額
A商店	テレビ	3	198,000	594,000
Bマート	テレビ	10	198,000	1,980,000
Bマート	洗濯機	5	59,800	299,000
C社	餅つき機	1	29,800	29,800

第2正規形の条件を満たす

情報無損失分解

注文[顧客名, 商品名, 数量, 金額]

顧客名	商品名	数量	金額
A商店	テレビ	3	594,000
Bマート	テレビ	10	1,980,000
Bマート	洗濯機	5	299,000
C社	餅つき機	1	29,800

注文[商品名, 単価]

商品名	単価
テレビ	198,000
洗濯機	59,800
餅つき機	29,800

第3 正規形

- テーブルが第2 正規形であり, テーブルの全ての非キー属性がいかなる候補キーにも推移的に関数従属しない状態
- 推移的な関数従属性: $A \rightarrow B$, $B \rightarrow C$ の場合 $A \rightarrow B \rightarrow C$ という関数従属性が存在
- 非キー属性間の関数従属性を引き離すように情報無損失分解を行う

情報無損失分解

社員

社員番号	社員名	給与	所属	勤務地
0650	鈴木一郎	50	K55	神奈川
1508	浜崎アユ	40	K41	東京
0231	宇田ひかる	60	K41	東京
2034	別所幸治	40	K55	神奈川
2100	森虎丸	40	K58	静岡

非キー属性間の関数従属性
 f_4 : 所属 → 勤務地 を使って
情報無損失分解

情報無損失分解

社員[社員番号, 社員名, 給与, 所属]

社員番号	社員名	給与	所属
0650	鈴木一郎	50	K55
1508	浜崎アユ	40	K41
0231	宇田ひかる	60	K41
2034	別所幸治	40	K55
2100	森虎丸	40	K58

社員[所属, 勤務地]

所属	勤務地
K41	東京
K55	神奈川
K58	静岡

E-R 図 (Entity-Relationship Diagram)

- 実体と関連を表す図. 関連では対応関係を記述する
 - 1 対 1 , 1 対多, 多対 1 , 多対多の関係がある
 - エンティティ間の共通の属性を見てエンティティ間にどのような対応関係があるかを考える

E-R 図の UML 表記

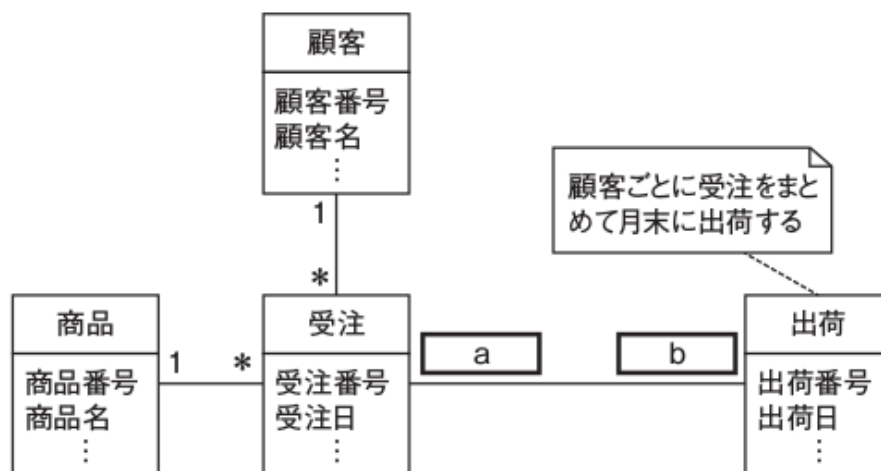
- E-R 図を UML を用いて表記することがある

多重度	意味
*	ゼロ以上の数字すべて
1.. *	1 以上
0..1	ゼロまたは 1
1	1
a..b (a, b は任意の数字)	a から b まで

問題

問題

その月に受注した商品を、顧客ごとにまとめて月末に出荷する場合、受注クラスと出荷クラスとの間の関連のa, bに入る多重度の組合せはどれか。ここで、出荷のデータは実績に基づいて登録される。また、モデルの表記にはUMLを用いる。



	a	b
ア	1	1..*
イ	1	0..1
ウ	1..*	0..1
エ	1..*	1..*

答え

解説

空欄aについて、出荷に対する受注は、注釈に「顧客ごとに受注をまとめて月末に出荷する」とあるので、複数の受注に対して出荷が一つになることが分かります。受注した商品を出荷するため受注は必ずあることになるので、多重度は1.＊です。

空欄bについて、受注に対する出荷は、問題文に「その月に受注した商品を、顧客ごとにまとめて月末に出荷」とあり、月の途中では、受注に対する出荷は存在しないことになります。したがって、受注に対する出荷はないこともあると考えられ、多重度は0.1となります。

これらを組み合わせると、ウが正解になります。

《解答》ウ

制約

- テーブル間の関係を維持するために制約を設定する必要がある
- 参照制約: 2 つのテーブルの関係の参照整合性を維持するために外部キーを設定する
- 一意性制約: データに空値を許さない制約
- 検査制約: データの範囲をなどの形式を制限する制約

問題

問題

次の表定義において，“在庫”表の製品番号に定義された参照制約によって拒否される可能性のある操作はどれか。ここで、実線は主キーを、破線は外部キーを表す。

在庫（在庫管理番号，製品番号，在庫量）

製品（製品番号，製品名，型，単価）

ア “在庫”表の行削除

イ “在庫”表の表削除

ウ “在庫”表への行追加

エ “製品”表への行追加

（平成22年秋 応用情報技術者試験 午前 問32）

答え

解説

在庫と製品の関連をE-R図に表すと、右のようになります。



このとき、“在庫”表に行を追加する場合と、“製品”表から行を削除する場合に参照制約がかかります。“製品”表にないものを“在庫”表に追加する、まだ在庫があるのに“製品”表の製品を削除する、などが起こると参照整合性に異状が発生するので、これを阻止します。したがって、ウが正解です。

《解答》ウ

パフォーマンス設計

- データベース設計を変更する
 - テーブル構造や関係を変えることによってパフォーマンスを向上させる. 代表的な方法として非正規化がある
- SQL のパフォーマンスのチューニングを行う
 - SQL を効率的なものに変更する
- DBMS の機能を使う
 - インデックスを設定したり, データベースを再編成してアクセス効率を上げたりする