計算機科学実験及演習4（データベース）課題5

1029-33-0786

松井 玲

1. **設計したER図**

ユーザーは名前と電話番号を持ち、ユーザー名は登録時に他ユーザーとの重複を許さないので、キー属性となる。

ユーザーは各レストランに対し評価（点数）を行い、さらにレストランはユーザーに対しクーポン（割引率）をもつ関連集合が存在する。評価点は0から100の間の整数とする。

レストランはID、名前、住所、そして電話番号を持ち、IDによって一意に定まる。

レストランは料理に応じて地域別にまとめられ、検索範囲の地域ごとのランキングを関連集合にもつ。ランキングは自然数である。

検索地域

評価

ユーザー

レストラン

評価-割引率

ランキング

1. **設計した関係スキーマ**

**関係：(ユーザー（ユーザー名、電話番号、レビュー数）,Σ（ユーザー）)**

Σ(ユーザー)は次の一貫性制約を含む

σ1：属性（ユーザー名）に含まれる値に重複はない

σ2：属性（電話番号）に含まれる値に重複はない

σ3：dom(ユーザー名) = String

σ4：dom(電話番号) = String

σ5：dom(レビュー数) = Integer

関数従属性集合F={ユーザー名→電話番号、レビュー数

電話番号→ユーザー名、レビュー数}

ユーザー名は各ユーザーに一つで、重複はない。また、電話番号も同様の性質を持つ。

**関係：(レストラン（レストランID、レストラン名、電話番号、住所）,Σ（レストラン）)**

Σ(レストラン)は次の一貫性制約を含む

σ1：属性集合{レストランID}が主キーである。

σ2：dom(レストランID) = Integer (1≦レストランID)

σ3：dom(レストラン名) = String

σ4：dom(電話番号) = String

σ5：dom(住所) = String

関数従属性集合F={レストランID→レストラン名、電話番号、住所

レストラン名、住所→レストランID、電話番号

電話番号→レストランID、レストラン名、住所）

レストランIDは各ユーザーに一つで、重複はない。また電話番号、レストラン名と住所の組も同様の性質を持つ。

**関係：(評価-割引率（ユーザー名、レストランID、評価、割引率）,Σ(評価-割引率）)**

Σ(評価-割引率)は次の一貫性制約を含む

σ1：属性集合{ユーザー名、レストランID}が主キーである。

σ2：dom(ユーザー名) = String

σ3：dom(評価) = Integer (0≦評価≦100)

σ4：dom(割引率) = Integer (0≦割引率≦100)

σ5：dom(レストランID) = Integer (1≦レストランID)

関数従属性集合F={ユーザー名、レストランID→評価、割引率}

ユーザー名とレストラン名の組が決まれば、そのユーザーがレストランに対して書いた評価も一つに定まる。

**関係：(ランキング（レストランID、地域名、ランキング）,Σ（ランキング）)**

Σ(ランキング)は次の一貫性制約を含む

σ1：属性集合{レストランID、地域名}が主キーである。

σ2：dom(レストランID) = Integer (1≦レストランID)

σ3：dom(地域名) = String

σ4：dom(ランキング) = Integer (1≦ランキング)

関数従属性集合F={レストランID、地域名→ランキング}

レストランIDと検索範囲の地域名が決まれば、その地域内でのランキングも一つに定まる。

これらの関係スキーマにおいて多値従属性は存在しない。

1. **関係スキーマの正規化**
   1. **関係：(ユーザー（ユーザー名、電話番号、レビュー数）,Σ（ユーザー）)**

Σ(ユーザー)は次の一貫性制約を含む

σ1：属性（ユーザー名）に含まれる値に重複はない

σ2：属性（電話番号）に含まれる値に重複はない

σ3：dom(ユーザー名) = String

σ4：dom(電話番号) = String

σ5：dom(レビュー数) = Integer

関数従属性集合F={ユーザー名→電話番号、レビュー数

電話番号→ユーザー名、レビュー数}

この関係の候補キーは属性(ユーザー名),(電話番号})である。

これは部分関数従属を持たず、データは値であるため第2正規形である。

決定項はともに候補キーであるのでボイスコッド正規形である。

* 1. **関係：(レストラン（レストランID、レストラン名、電話番号、住所）,Σ（レストラン）)**

Σ(レストラン)は次の一貫性制約を含む

σ1：属性集合{レストランID}が主キーである。

σ2：dom(レストランID) = Integer (1≦レストランID)

σ3：dom(レストラン名) = String

σ4：dom(電話番号) = String

σ5：dom(住所) = String

関数従属性集合F={レストランID→レストラン名、電話番号、住所

レストラン名、住所→レストランID、電話番号

電話番号→レストランID、レストラン名、住所）}

この関係の候補キーは属性{レストランID}、{電話番号}、{レストラン名、住所}である。

よってこの関係は部分関数従属を持たず、決定項は候補キーであるのでボイスコッド正規形である。

* 1. **関係：(評価-割引率（ユーザー名、レストランID、評価、割引率）,Σ(評価-割引率）)**

Σ(評価-割引率)は次の一貫性制約を含む

σ1：属性集合{ユーザー名、レストランID}が主キーである。

σ2：dom(ユーザー名) = String

σ3：dom(評価) = Integer (0≦評価≦100)

σ4：dom(割引率) = Integer (0≦割引率≦100)

σ5：dom(レストランID) = Integer (1≦レストランID)

関数従属性集合F={ユーザー名、レストランID→評価、割引率}

この関係の決定項は主キーであるので、これはボイスコッド正規形を満たしている。

* 1. **関係：(ランキング（レストランID、地域名、ランキング）,Σ（ランキング）)**

Σ(ランキング)は次の一貫性制約を含む

σ1：属性集合{レストランID、地域名}が主キーである。

σ2：dom(レストランID) = Integer (1≦レストランID)

σ3：dom(地域名) = String

σ4：dom(ランキング) = Integer (1≦ランキング)

関数従属性集合F={レストランID、地域名→ランキング}

この決定項は主キーであるのでボイスコッド正規形を満たしている。

1. **正規化の手法**

3NFやBCNFでない設計からBCNFに変換するためには、以下のステップとアプローチが考えられる。

1. **候補キーの確認:**
   * **BCNFに変換するためには、まずテーブル内の候補キーを確認する。候補キーは、一意性制約を満たす属性または属性の組み合わせである。**
2. **関数従属性の特定:**
   * **テーブル内の関数従属性を特定する。**
3. **部分関数従属性の解決:**
   * **テーブル内に部分関数従属性が存在する場合、それらを解決する。これを行うためには、新しいテーブルを作成し、部分関数従属性の属性を移動する。**
4. **関数従属性の分解:**
   * **BCNFに変換するために、関数従属性を分解する、つまり関数従属性を満たす新しいテーブルを作成する。**
   * **次の条件を満たす属性集合X,Y,Z()を選ぶ**
     1. **XYZ = V**
     2. **Σ X→Y**
     3. **各AZに対してΣ X→A**
   * **属性集合を(XY),(XZ)に分割し、R中にWW’なる二つの関係スキーマが存在すればWを削除する**

この手法を用い続ければBCNFに変換することができる。

1. **テーブルの定義とデータの挿入**
2. **表の定義において，キーの指定により保持できる関数従属性や正規形について考察しなさい．表を正規化していく（例えば分解法や合成法）中で，注目するキーや関数従属性で結果がどのように変化するか（どの正規形になるか，どの関数従属性が保持されるか）を考察しなさい．**

　まず、分解法は上記4で述べた正規化の手法であり、BCNFまで正規化することができるが、テーブルの分解の過程で関数従属性が保持されない場合がある。X→Yが存在するにもかかわらずXとYのテーブルを分割してしまう場合があるからである。

一方で合成法は各関数従属性の両辺を関係スキーマとし、元の関係スキーマのキーを含む関係があれば終了する。この手法は関数従属性を保持できるが第３正規形までの合成しか保証できない。

1. **課題３で設計した関係スキーマに基づいて関係表を定義しなさい．定義するためのSQL文を示しなさい．なお，各関数従属性が保持されることを文章で説明しなさい。**

**関係：(ユーザー（ユーザー名、電話番号、レビュー数）,Σ（ユーザー）)**

関数従属性集合F={ユーザー名→電話番号、レビュー数

電話番号→ユーザー名、レビュー数}

CREATE TABLE Users (

UserName TEXT NOT NULL,

PhoneNumber TEXT NOT NULL,

ReviewCount INTEGER,

PRIMARY KEY (UserName, PhoneNumber),

UNIQUE (UserName),

UNIQUE (PhoneNumber)

);

**図１：Usersテーブルの定義文**

**関係：(レストラン（レストランID、レストラン名、電話番号、住所）,Σ（レストラン）)**

関数従属性集合F={レストランID→レストラン名、電話番号、住所

レストラン名、住所→レストランID、電話番号

電話番号→レストランID、レストラン名、住所）}

CREATE TABLE Restaurants (

RestaurantID INTEGER PRIMARY KEY,

RestaurantName TEXT,

PhoneNumber TEXT,

Address TEXT

);

図2：Restaurantsテーブルの定義文

**関係：(評価-割引率（ユーザー名、レストランID、評価、割引率）,Σ(評価-割引率）)**

関数従属性集合F={ユーザー名、レストランID→評価、割引率}

CREATE TABLE RatingsDiscounts (

UserName TEXT,

RestaurantID INTEGER,

Rating INTEGER,

DiscountRate INTEGER,

PRIMARY KEY (UserName, RestaurantID),

FOREIGN KEY (UserName) REFERENCES Users(UserName),

FOREIGN KEY (RestaurantID) REFERENCES Restaurants(RestaurantID)

);

図３：RatingDiscountsテーブルの定義文

**関係：(ランキング（レストランID、地域名、ランキング）,Σ（ランキング）)**

関数従属性集合F={レストランID、地域名→ランキング}

CREATE TABLE Rankings (

RestaurantID INTEGER,

RegionName TEXT,

Ranking INTEGER,

PRIMARY KEY (RestaurantID, RegionName),

FOREIGN KEY (RestaurantID) REFERENCES Restaurants(RestaurantID)

);

図４：Rankingsテーブルの定義文

これらのSQL文で関係表を定義することができる。

また、関係従属性集合に含まれる関数従属性は分割されていないため、保存されている。

1. **データを作成して，上記の表に挿入しなさい．データを挿入するためのSQL文を示しなさい．また，データを挿入した表の出力（先頭の一部で結構）を示しなさい．**

以下にデータ挿入文と挿入結果（の一部）を示す

INSERT INTO Users (UserName, PhoneNumber, ReviewCount)

VALUES

('田中 太郎', '555-1111', 5),

('山本 さちこ', '555-2222', 8),

('佐藤 一郎', '555-3333', 12),

('鈴木 由美', '555-4444', 7),

('高橋 健太', '555-5555', 3),

('渡辺 朋子', '555-6666', 10),

('斎藤 裕太', '555-7777', 6),

('伊藤 美穂', '555-8888', 9),

('中村 雄大', '555-9999', 15),

図5：Usersテーブルにデータ挿入文

グラフィカル ユーザー インターフェイス, テキスト, アプリケーション

自動的に生成された説明図6：Usersテーブルの挿入結果

INSERT INTO Restaurants (RestaurantName, PhoneNumber, Address)

VALUES

('居酒屋 たんか', '03-1111-2222', '東京都新宿区1-1-1'),

('寿司 まぐろや', '06-2222-3333', '大阪府大阪市中央区2-2-2'),

('焼肉 こだわり家', '075-3333-4444', '京都府京都市東山区3-3-3'),

('ラーメン 一八', '092-4444-5555', '福岡県福岡市博多区4-4-4'),

('カフェ ゆり', '052-5555-6666', '愛知県名古屋市中村区5-5-5'),

('蕎麦 かずのこ', '011-6666-7777', '北海道札幌市中央区6-6-6'),

('うどん やまと', '082-7777-8888', '広島県広島市中区7-7-7'),

図7：Restaurantsテーブルのデータ挿入文

グラフィカル ユーザー インターフェイス, テキスト, アプリケーション

自動的に生成された説明

図8：Restaurantsテーブルの挿入結果

INSERT INTO RatingsDiscounts (UserName, RestaurantID, Rating, DiscountRate)

VALUES

('田中 太郎', 1, 90, 10),

('山本 さちこ', 2, 85, 15),

('佐藤 一郎', 3, 78, 12),

('鈴木 由美', 4, 92, 8),

図9：RatingDiscountsテーブルのデータ挿入文

グラフィカル ユーザー インターフェイス, テキスト, アプリケーション

自動的に生成された説明

図10：RatingDiscountsテーブルの挿入結果

INSERT INTO Rankings (RestaurantID, RegionName, Ranking) VALUES

(1, '関東', 1),

(11, '関東', 3),

(2, '関西', 2),

(12, '関西', 1),

(3, '関西', 3

図11：Rankingsテーブルのデータ挿入文

グラフィカル ユーザー インターフェイス, テキスト, アプリケーション

自動的に生成された説明

図12：Rankingsテーブルの挿入結果

1. 課題5について
2. 関係代数の射影および選択に対応するSQL文

SELECT UserName, PhoneNumber, ReviewCount

FROM Users

WHERE ReviewCount >= 10;

図13：レビュー数が10以上のユーザーを選択する文

これはUsersテーブルからレビュー数が10以上のユーザーを選ぶ選択文であり、より評価を行なっている信頼性の高いユーザーを見つける時に使えると考えられる。

以下は実行結果である。

パソコンの画面

中程度の精度で自動的に生成された説明

図１４：選択の実行結果

1. 関係代数の自然結合に対応するSQL文

SELECT \*

FROM Restaurants

NATURAL JOIN RatingsDiscounts;

　図１５：レストランと評価割引率の自然結合

上記はRestaurantsテーブルとRatingDiscountテーブルの自然結合であり、各レストランとユーザーの組に対してその評価と付与される割引率が一目でわかるという点で有用である。

以下はその実行結果である。

パソコンの画面

中程度の精度で自動的に生成された説明

図１６：自然結合の実行結果

1. UNIONを含むSQL文

SELECT UserName

FROM RatingsDiscounts

WHERE Rating >= 95

UNION

SELECT UserName

FROM RatingsDiscounts

WHERE Rating <= 20;

これはUNION文を用いて極端に評価が高いか低いユーザーの名前を表示する。テキスト が含まれている画像

自動的に生成された説明

以上は実行結果である。

1. EXCEPT文を含むSQL文

SELECT UserName

FROM RatingsDiscounts

EXCEPT

SELECT UserName

FROM RatingsDiscounts

WHERE Rating <= 20;

上記は20点以下の極端に低い評価をつけたユーザー名を排除して表示するSQL文であり、荒らし目的のユーザーを排除する際に有用であると考えられる。

以下は実行結果である。

テキスト, スコアボード, 画面, テレビ が含まれている画像

自動的に生成された説明

1. DISTIINCTを含むSQL文

SELECT DISTINCT UserName

FROM RatingsDiscounts

上記はDISTINCT文を用いて評価を行なったユーザー名を重複なしで取り出すことができ、評価を行なったユーザーを一目で見ることができる。

以下は実行結果である。

テキスト, スコアボード が含まれている画像

自動的に生成された説明

1. AVGを用いたSQL文

SELECT AVG(Rating),RestaurantName

FROM (RatingsDiscounts NATURAL JOIN Restaurants)

WHERE RestaurantID = 5

上記はRestaurantIDが5のレストランの名前と平均の評価を表示するSQL文である。

以下は実行結果である。

黒い背景に白い文字がある

低い精度で自動的に生成された説明

1. 副質問を含むSQL文

SELECT RestaurantName

FROM Restaurants

WHERE RestaurantID = (

SELECT RestaurantID

FROM (

SELECT RestaurantID, AVG(Rating) AS AvgRating

FROM RatingsDiscounts

GROUP BY RestaurantID

) AS Subquery

WHERE AvgRating = (

SELECT MAX(AvgRating)

FROM (

SELECT AVG(Rating) AS AvgRating

FROM RatingsDiscounts

GROUP BY RestaurantID

) AS MaxSubquery

)

);

これは副質問を使い平均の評価が最も高いレストラン名を表示するSQL文である。

まず内部のサブクエリを使用して、各レストランの平均評価を計算する。次に、もう一つのサブクエリを使用して、最も高い平均評価を持つ **RestaurantID** を見つけ、最後に、その **RestaurantID** に対応するレストラン名を取得する。

以下は実行結果である。

グラフィカル ユーザー インターフェイス

自動的に生成された説明

1. UPDATEを含むSQL文

UPDATE Users

SET PhoneNumber = '555-2929'

WHERE UserName= '田中 太郎';

上記はユーザーの電話番号を変更するUPDATE文であ。

以下は実行結果である。



1. ORDER BYを含むSQL文

SELECT Restaurants.RestaurantName, AVG(RatingsDiscounts.Rating) AS AverageRating

FROM Restaurants

JOIN RatingsDiscounts ON Restaurants.RestaurantID = RatingsDiscounts.RestaurantID

GROUP BY Restaurants.RestaurantName

ORDER BY AverageRating DESC;

上記はレストランごとの平均評価を高い順に表示するORDER BYを用いた文である。

以下はその実行結果である。

記号 が含まれている画像

自動的に生成された説明

1. CREATE VIEWを含むSQL文

CREATE VIEW AverageRatings AS

SELECT Restaurants.RestaurantName, AVG(RatingsDiscounts.Rating) AS AverageRating

FROM Restaurants

JOIN RatingsDiscounts ON Restaurants.RestaurantID = RatingsDiscounts.RestaurantID

GROUP BY Restaurants.RestaurantName;

上記は平均評価をAverageRatingsとして表にするSQL文である。これを用いると今までの評価を用いる文が書きやすくなる。

以下は実行結果である。

黒い背景と白い文字

自動的に生成された説明