データベースシステム 129－18
（２００３．１．２４）

# セルラーモデルに基づいたセルラーＤＢＭＳの開発

## -RDB のマスターデータ問題への対応-

児玉　敏男[†1]　　國井　利泰[†2]

**要旨**

既存の汎用データベースであるリレーショナル DBMS はワールドモデルとしてデータの依存関係全体を把握しているデータベース管理者の存在を前提にしており，テーブルのデータとその依存関係が動的に変化する状況へのシステム適用は非常に困難である．さらに，リレーショナル DBMS が数学的に最も抽象的な集合論に基づいたリレーショナルモデルを採用しているために，例えば部分集合を要素として扱うトポロジー的処理はモデル外であり，同値関係もモデル外であり，データの変化を保存するホモトピー保存機能もモデル外である等，モデルレベルでの限界が多い．本論文では，リレーショナル DBMS 運用時にマスターテーブルがもたらす問題点を，システム運用現場へのヒアリング調査によって抽出し 7 つのパターンに分類した．それらの問題点は，1. インスタンス変化のホモトピーが保存されないこと，2. インスタンス間の同値関係による対応がとれないこと，3. null インスタンスの概念がないこと，が原因である．そこで，優れた情報空間構築モデルであるセルモデルを採用したセルラーDBMS を開発し，上記の問題点がセルラーDBMS では全て解決可能であることを事例によって示した．

# Development of a Cellular DBMS based on a Cellular Model

## - Solution of the master data problems of the RDB -

Toshio Kodama[†1]　　Tosiyasu L. Kunii[†2]

**Abstract**

A popular commercial DBMS, the relational DBMS is built under the assumption of the existence of a database administrator who knows all the dependencies of the data responsible to manage.　It is difficult to adapt it to the situation where data and its dependencies are dynamically changing. The relational DBMS applies the relational model that is based on the set theory which is most abstract mathematically; for example topology processing which treats a subset as an element is out of its model, an equivalence relation is also out of its model and a homotopy preservation function which preserves data changes is also out of its model, etc. Thus, there are too many limits in the relational data model at its model level. We have isolated the problems with managing a master table of the relational DBMS by extensive investigation with people who have been actually engaged in the operation of the large scale relational system administration and have classified them into 7 patterns.　It turns out that the reasons for the problems are:(1) homotopy of instance changes isn't preserved, (2) equivalence relations aren't applied between instances, (3) there is no concept of null instances. We have developed the cellular model to model web-based information spaces for designing the cellular DBMS. We demonstrate that the cell DBMS solves all the problems listed above.

## 1.　はじめに

インターネット上で, 様々なコンピュータ機器がグローバルに常時接続される時代に入り, 莫大な情報が常時処理されるようになった．その中，各企業が展開するインターネットモールでの顧客管理, コンビニエンスストアでの商品の在庫管理，建設工事現場での資機材管理等々, データ管理の前提となる要求項目が常に変化する状況に適合した情報管理のしくみが必要とされるようになった．しかし，現在，最も汎用的に利用されているデータベースであるリレーショナル DBMS を振り返ると，データの依存関係が変化せず，変化に対してはデータベース管理者の存在が必要であるという静的状況を前提にしている．そこで，優れた情報結合モデルであるセルモデルを応用し, データの依存関係が動的に変化するという現在の状況に柔軟に対応できる新しいデータベースの開発を本研究の目的とする．本稿では特に，リレーショナルＤＢＭＳにおけるマスターテーブルがもたらす問題点が，ホモトピー理論・同値関係による対応が定義されたセルラーDBMS では解決可能であることを具体的に例を上げて示す．

本稿は以下のように構成される．第２章では，ヒアリング調査に基づき，既存のリレーショナル DBMS のマスターテーブルが抱える問題点を述べる．第３章では，セルラーDBMS の定義を行い, 第４章でそれらのマスターテーブルの問題点がセルラーDBMS によって解決可能であることを具体的な事例を上げて述べる. 第５章では結論を述べる．

†1　前田建設工業（株）
Maeda Corporation
†2　法政大学大学院情報科学研究科
Graduate School of Computer and Information Sciences, Hosei University

## 2. リレーショナル DBMS のマスターテーブルがもたらす問題点

リレーショナルＤＢの正規化において, インスタンスは変化しないことを前提におくマスターテーブルと事象の発生を記録するトランザクションテーブルに明確に区別される．しかし，現実的にはマスターテーブルのインスタンスは様々に変化する場合が多く, そのときの対応には要求に応じてアプリケーションレベルで様々な工夫が必要となり非常に困難な場合が多い. そのようなマスターテーブルがもたらす問題点について情報系企業数社にヒアリング調査を行った．その結果，問題点は以下に述べる 7 点であることが分かった．

- あるインスタンスが他のインスタンスに変化したがあったときの対応が難しい

例えば，企業 A の属性情報“本社所在地＝a”が移転により“本社所在地＝b”へ変更されたときの表現が難しい．

- 複数のインスタンスが結合して，新たなインスタンスが生成されたときの対応が難しい

例えば，企業 A と企業 B が合併して企業 C になるという変化を表現することが難しい．

- あるインスタンスが分解して，複数のインスタンスが生成されたときの対応が難しい

例えば，企業 C 内の一部署が企業 D として分社化したという変化を表現することが難しい．

これらは, リレーショナルＤＢが採用するリレーショナルモデルは数学的に最も抽象度の高い集合論を基礎においているのでインスタンスの変化のホモトピーを表現する機構が存在しないことが原因である．

- あるインスタンスが発生したときの対応が難しい

例えば，企業 E が新たに設立されたという表現をすることが難しい．

- あるインスタンスが消滅したときの対応が難しい

例えば, 企業 F が倒産して無くなったという表現をすることが難しい．

これらは, リレーショナルモデルには前述のホモトピーを表現する機構が無いことに加えて，null インスタンスという概念が無いことが原因である．

- マスターテーブルのキー属性のデータ重複への対応が難しい．
- 同じ実体に対して異なるインスタンスを作成したときの対応が難しい．

これらは, リレーショナルＤＢが採用するリレーショナルモデルにはインスタンスの属性データ間で同値関係による対応を表現する機構がないことが主な原因である．

## 3. セルラーDBMS の定義

セルラーモデルは，著者の一人（國井）によって 1999 年に発表されたモデルで, 現実のローカルな情報とグローバルな情報の結合が可能で，現実世界・サイバー世界の全ての事象を射影可能な優れた情報空間構築モデルである．[3][4][5]

セルラーモデルを数学的基礎においたセルラー DBMS は，1．セル情報，2．セル定義情報，3．セル操作情報の 3 つの情報から構成されるとする．

### 3.1 セル情報

セル情報には，各セルを一意に識別するセル ID，セルの属性とそのタプルを記録する．セルは，リレーショナル DB のテーブルに相当し，セル ID はリレーショナルモデルのテーブル ID に相当する．また，セルの属性とタプルは，変数と値の関係にあり，2 次元の表として表現されるとする．

また, DBMS が全体のセルを管理するための管理セルが設けられ，そこでセル ID が定義される．

### 3.2 セル定義情報

セル定義情報には, 各セルのオープンセルと境界の情報を記録する．

セルモデルでは，境界があるセルをクローズドセル，境界が無いセルをオープンセルという．

すなわち，クローズドセル $B^n$ のオープンセル $e^n$, 境界を $\partial B^n$ とすると

$$\partial B^n = B^n - e^n$$

セルラーDBMS において, オープンセルはクローズドセルに含まれる 1 つの属性であると同時に, オープンセル自体も属性数が 1 のセルとして境界と分離されて取り扱われる．また，オープンセルはリレーショナル DB のプライマリーキーに相当する役割を果たし, クローズドセルのタプルを一意に識別する 1 つの属性が選定される．

このように，クローズドセルとそのオープンセル・境界の情報を記録することにより, セルのデータ管理が非常に容易になる．例えば，オープンセルと境界を分離されうることで, 各セルのオープンセルのみを取り出すことで, クライアント側に負担のかからないデータ転送量の小さなインデックス機能として活用することが可能である．また，複数のセルがセル接合した状態からデータ出力を行うとき，データの全属性を指定しなくとも，記録されたクローズドセルの情報を指定すればユーザー要求に応えることができる. つまりオブジェクト単位でのデータ管理を行うことができる．

### 3.3 セル操作情報

セル操作情報には，スキーマレベル・インスタンスレベルの両レベルにおいて, セルの変化のホモトピーに関する操作情報を記録する. 保存されたホモトピーによりセルの変化を追跡することができるので, 現実の事象の変化をきちんと表現することができる. またセルの変化のホモトピーを保存することでホモトピー同値性が保証されるので, 変化したセルを変形前の状態に戻すことができ, 変形前のセルを再び資源として活用する事がで

きる．つまりセルの再利用が可能になる．

## 4. 事例研究

2 章で上げたリレーショナル DBMS のマスターテーブルが抱える問題点に対して, セルモデルを採用したセルラーDBMS では解決可能であることを事例により示す.

### 4.1 テーブル、セルの設定

ある企業が取引先の企業情報とその製品の取引記録を管理するというシチュエーションにおいて、リレーショナル DBMS、セルラーDBMS で以下のようなデータベース設計を行う。

**リレーショナル** DBMS

企業情報を管理する企業マスターテーブル *R*(表 4.1.1, 主キー=”企業コード”）と取引記録を管理する取引記録トランザクションテーブル *S*（表 4.1.2, 主キー=”取引番号”）を設ける。この２つのテーブルは属性”企業コード”で関連を持ち JOIN 結合を行う。（表 4.1.3）

表 4.1.1　企業マスターテーブル　*R*

| 企業コード | 企業名 | 本社所在地 |
|---|---|---|
| C1 | A | 東京 |
| C2 | B | 大阪 |
| C3 | C | 愛知 |

表 4.1.2　取引記録テーブル *S*

| 取引番号 | 製品番号 | 企業コード |
|---|---|---|
| 122 | a | C1 |
| 123 | b | C1 |
| 124 | c | C2 |
| 125 | d | C3 |

表 4.1.3　JOIN 結合後のテーブル　*R*×*S*

| 取引番号 | 製品番号 | 企業コード | 企業名 | 本社所在地 |
|---|---|---|---|---|
| 122 | a | C1 | A | 東京 |
| 123 | b | C1 | A | 東京 |
| 124 | c | C2 | B | 大阪 |
| 125 | d | C3 | C | 愛知 |

**セルラー**DBMS

同様にして, クローズドセルとして企業セル$\mathcal{B}_c^4$,（表 4.1.4），取引記録セル$\mathcal{B}_p^4$（表 4.1.6）を設ける．このときセルのインスタンスが追加される毎に自動付番される属性 obj-id を各セル（表 4.1.5,4.1.7）に設ける．この属性はオープンセルと呼ばれ, セルのインスタンスを一意に特定する属性数 1 のセルである．[1]

また同値関係により 2 つのセルを接合して統合セルが作成される．（表 4.1.8，図 4.1）

表 4.1.4　企業セル　$\mathcal{B}_c^4$

| obj-id_1 | 企業コード | 企業名 | 本社所在地 |
|---|---|---|---|
| 1 | C1 | A | 東京 |
| 2 | C2 | B | 大阪 |
| 3 | C3 | C | 愛知 |

表 4.1.5　企業セル(オープン)　$e_c^1$

| obj-id_1 |
|---|
| 1 |
| 2 |
| 3 |

表 4.1.6 取引記録セル　$\mathcal{B}_p^4$

| obj-id_2 | 取引番号 | 製品番号 | obj-id_1 |
|---|---|---|---|
| 1 | 122 | a | 1 |
| 2 | 123 | b | 1 |
| 3 | 124 | c | 2 |
| 4 | 125 | d | 3 |

表 4.1.7 取引記録セル(オープン)　$e_p^1$

| obj-id_2 |
|---|
| 1 |
| 2 |
| 3 |
| 4 |

セル分解・接合による統合セルの作成

企業セルの分解写像　c:

$\mathcal{B}_c^4 \rightarrow \mathcal{B}_c^1$（obj-id_1）$\sqcup_g \mathcal{B}_c^3$

アイデンティフィケーション関数 g:

$\partial\partial\partial\mathcal{B}_c^4 \rightarrow \mathcal{B}_c^1$（obj-id_1）

取引記録セルの分解写像　a:

$\mathcal{B}_p^4 \rightarrow \mathcal{B}_p^1$（obj-id_1）$\sqcup_h \mathcal{B}_p^3$

アイデンティフィケーション関数 h:

$\partial\partial\partial\mathcal{B}_p^4 \rightarrow \mathcal{B}_p^1$（obj-id_1）

セル接合写像　k:

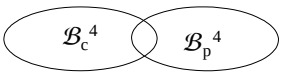
$\mathcal{B}_c^1$（obj-id_1）$\rightarrow \mathcal{B}_p^1$（obj-id_1）

作成された統合セル

$\mathcal{B}_c^4 \sqcup_k \mathcal{B}_p^4 = \mathcal{B}_c^4 \sqcup \mathcal{B}_p^4 /\sim$

表 4.1.8 統合セル　$\mathcal{B}_c^4 \sqcup \mathcal{B}_p^4 /\sim$

| obj-id_2 | 取引番号 | 製品番号 | obj-id_1 | 企業コード | 企業名 | 本社所在地 |
|---|---|---|---|---|---|---|
| 1 | 122 | a | 1 | C1 | A | 東京 |
| 2 | 123 | b | 1 | C1 | A | 東京 |
| 3 | 124 | c | 2 | C2 | B | 大阪 |
| 4 | 125 | d | 3 | C3 | C | 愛知 |




$\mathcal{B}_c^4 \sqcup \mathcal{B}_p^4 / \sim$

図 4.1　作成された統合セル

### 4.2 セルラーDBMS によるマスターテーブルが抱える問題点の解決

設定されたテーブル・セルに対して，6 つの異なる事象が発生したとき，リレーショナル DBMS とセルラーDBMS での対応を事例 1～6 として上げ比較する．

**事例 1）インスタンスの変更**

事象「ある時，企業コード＝”C1”の企業が名称変更によ

り，A から A’になった．その後，継続してその企業が製品番号 a の製品に対し取引を行った.」

**リレーショナル** DBMS **での対応**

リレーショナル DBMS では属性情報の変化の表現は不可能である．

手順として，一般的には企業コード=”C1”の企業名を A から A’に更新する．

表 4.2.1　企業マスターテーブル　$R$

| 企業コード | 企業名 | 本社所在地 |
|---|---|---|
| C1 | **A’** | 東京 |
| C2 | B | 大阪 |
| C3 | C | 愛知 |

また取引記録として取引記録テーブルに新しいインスタンスを追加する．

表 4.2.2　取引記録テーブル $S$

| 取引番号 | 製品番号 | 企業コード |
|---|---|---|
| 122 | a | C1 |
| 123 | b | C1 |
| 124 | c | C2 |
| 125 | d | C3 |
| **126** | **a** | **C1** |

表 4.2.3　JOIN 結合後のテーブル　$R \times S$

| 取引番号 | 製品番号 | 企業コード | 企業名 | 本社所在地 |
|---|---|---|---|---|
| 122 | a | C1 | **A’** | 東京 |
| 123 | b | C1 | **A’** | 東京 |
| 124 | c | C2 | B | 大阪 |
| 125 | d | C3 | C | 愛知 |
| 126 | a | C1 | **A’** | 東京 |

この結合後のテーブルでは，取引番号=126 のインスタンスにおいて，企業コード=”C1”の属性である企業名は A’となり最新情報はきちんと表現されているが，変更前の過去の取引である取引番号=122,123 のインスタンスでも企業名が A’であり，事象をきちんと表現したことにならない．

**セルラー**DBMS **での対応**

セルラーDBMS では企業の属性情報の変化を表現可能である．

手順として，企業名＝“ A’ ”の新しいインスタンス（obj-id_1=4）を追加し（表 4.2.1），そして企業名の変化のホモトピーを保存する．

表 4.2.4　変更後の企業セル　$\mathcal{B}_c^4$

| obj-id_1 | 企業コード | 企業名 | 本社所在地 |
|---|---|---|---|
| 1 | C1 | A | 東京 |
| 2 | C2 | B | 大阪 |
| 3 | C3 | C | 愛知 |
| 4 | C1 | **A’** | 東京 |

表 4.2.5　変化後の企業セル(オープン)　$e_c^1$

| obj-id_1 |
|---|
| 1 |
| 2 |
| 3 |
| 4 |

企業セル(オープン)において変化のホモトピーh を保存する．

$$h\colon 1 \to 4$$

表 4.2.6 取引記録セル　$\mathcal{B}_p^4$

| obj-id_2 | 取引番号 | 製品番号 | obj-id_1 |
|---|---|---|---|
| 1 | 122 | a | 1 |
| 2 | 123 | b | 1 |
| 3 | 124 | c | 2 |
| 4 | 125 | d | 3 |
| 5 | 126 | a | 4 |

表 4.2.7 統合セル　$\mathcal{B}_c^4 \sqcup \mathcal{B}_p^4/\sim$

| obj -id_2 | 取引番号 | 製品番号 | obj -id_1 | 企業コード | 企業名 | 本社所在地 |
|---|---|---|---|---|---|---|
| 1 | 122 | a | 1 | C1 | **A** | 東京 |
| 2 | 123 | b | 1 | C1 | **A** | 東京 |
| 3 | 124 | c | 2 | C2 | B | 大阪 |
| 4 | 125 | d | 3 | C3 | C | 愛知 |
| 5 | 126 | a | 4 | C1 | **A’** | 東京 |

この統合セルと保存されたホモトピーより，企業コード=”C1”の属性である企業名の A から A’へ変更がきちんと表現されていることが分かる．

**事例** 2）**インスタンス間の結合**

事象「ある時，２つの企業（企業コード＝”C1,2”）が合併し，新企業（企業コード=”C4”，企業名=”D”，本社所在地=”東京”）が設立された．その後，その新企業が継続して製品番号=”a”の製品に対し取引を行った.」

**リレーショナル** DBMS **での対応**

リレーショナル DBMS では企業合併の表現は不可能である．

手順として，一般的には企業コード=”C4”として新しいインスタンスを追加する．

表 4.2.8　企業マスターテーブル　$R$

| 企業コード | 企業名 | 本社所在地 |
|---|---|---|
| C1 | A | 東京 |
| C2 | B | 大阪 |
| C3 | C | 愛知 |
| **C4** | **D** | **東京** |

ここでは，新たにインスタンスが登録されただけで２企業が合併するという変化を表現していない．

また，取引記録として取引記録テーブルに新しいインスタンスを追加する．

表 4.2.9　取引記録テーブル $S$

| 取引番号 | 製品番号 | 企業コード |
|---|---|---|
| 122 | a | C1 |
| 123 | b | C1 |
| 124 | c | C2 |
| 125 | d | C3 |
| **126** | **a** | **C4** |

表 4.2.10　JOIN 結合後のテーブル　$R \times S$

| 取引番号 | 製品番号 | 企業コード | 企業名 | 本社所在地 |
|---|---|---|---|---|
| **122** | **a** | **C1** | **A** | **東京** |
| **123** | **b** | **C1** | **A** | **東京** |
| **124** | **c** | **C2** | **B** | **大阪** |
| 125 | d | C3 | C | 愛知 |
| **126** | **a** | **C4** | **D** | **東京** |

この結合後のテーブルでは, 追加されたインスタンス（取引番号=126）での合併後の新企業（企業コード=”C4”）の取引はきちんと表現されているが，企業の合併という変化は表現されていないので, 合併前の２つの企業（企業コード=”C1,2”）の取引（取引番号=122,123,124）との関係は当然表現することはできない.

**セルラー**DBMS **での対応**

セルラーDBMS では企業合併の変化を表現することができる.

手順として，企業名＝”C4”の新しいインスタンス（obj-id_1=4）を追加し，そして企業が合併し新企業ができるという変化のホモトピーを保存する.

表 4.2.11　変更後の企業セル　$\mathcal{B}_c^4$

| obj-id_1 | 企業コード | 企業名 | 本社所在地 |
|---|---|---|---|
| 1 | C1 | A | 東京 |
| 2 | C2 | B | 大阪 |
| 3 | C3 | C | 愛知 |
| **4** | **C4** | **D** | **東京** |

表 4.2.12　変化後の企業セル(オープン)　$e_c^1$

| obj-id_1 |
|---|
| 1 |
| 2 |
| 3 |
| 4 |

企業セル(オープン)において変化のホモトピーh1,h2は

$$h1: 1 \to 4$$

$$h2: 2 \to 4$$

また, 取引記録として取引記録テーブルに新しいインスタンスを追加する.

表 4.2.13 取引記録セル　$\mathcal{B}_p^4$

| obj-id_2 | 取引番号 | 製品番号 | obj-id_1 |
|---|---|---|---|
| 1 | 122 | a | 1 |
| 2 | 123 | b | 1 |
| 3 | 124 | c | 2 |
| 4 | 125 | d | 3 |
| 5 | 126 | a | 4 |

このとき統合セルは以下になる。

表 4.2.14 統合セル　$\mathcal{B}_c^4 \sqcup \mathcal{B}_p^4/\sim$

| obj -id_2 | 取引番号 | 製品番号 | obj -id_1 | 企業コード | 企業名 | 本社所在地 |
|---|---|---|---|---|---|---|
| **1** | **122** | **a** | **1** | **C1** | **A** | **東京** |
| **2** | **123** | **b** | **1** | **C1** | **A** | **東京** |
| **3** | **124** | **c** | **2** | **C2** | **B** | **大阪** |
| 4 | 125 | d | 3 | C3 | C | 愛知 |
| **5** | **126** | **a** | **4** | **C4** | **D** | **東京** |

この統合セルと保存されたホモトピーより, 合併前の２企業（企業コード=”C1,2”）の取引（取引番号=122,123,124）と合併後の新企業（企業コード=”C4”）の取引（取引番号=126）の関係がきちんと表現されていることが分かる.

**事例 3）インスタンスの分解**

事象「ある時，１つの企業（企業コード＝”C1”）が企業分割をおこない２つの企業（企業コード=”C5,6” 企業名=”E,F”，本社所在地=”東京，神奈川”）になった．その後，その２企業がそれぞれ製品番号=”a, b”の製品に対し取引を行った.」

**リレーショナル** DBMS **での対応**

リレーショナル DBMS では企業分割の変化の表現は不可能である.

手順として, 一般的には企業コード=”C5,C6”として新しいインスタンスを追加する.

表 4.2.15　企業マスターテーブル　$R$

| 企業コード | 企業名 | 本社所在地 |
|---|---|---|
| C1 | A | 東京 |
| C2 | B | 大阪 |
| C3 | C | 愛知 |
| **C5** | **E** | **東京** |
| **C6** | **F** | **神奈川** |

ここでは, 新たに２つのインスタンスが登録されただけで企業が分割するという変化を表現していない.

また, 取引記録として取引記録テーブルに新しいインスタンスを追加する.

表 4.2.16　取引記録テーブル $S$

| 取引番号 | 製品番号 | 企業コード |
|---|---|---|
| 122 | a | C1 |
| 123 | b | C1 |
| 124 | c | C2 |
| 125 | d | C3 |
| **126** | **a** | **C5** |
| **127** | **b** | **C6** |

このとき JOIN 結合後のテーブルは以下になる。

表 4.2.17　JOIN 結合後のテーブル　$R \times S$

| 取引番号 | 製品番号 | 企業コード | 企業名 | 本社所在地 |
|---|---|---|---|---|
| **122** | **a** | **C1** | **A** | **東京** |
| **123** | **b** | **C1** | **A** | **東京** |
| 124 | c | C2 | B | 大阪 |
| 125 | d | C3 | C | 愛知 |
| **126** | **a** | **C5** | **E** | **東京** |
| **127** | **b** | **C6** | **F** | **神奈川** |

この結合後のテーブルでは, 追加されたインスタンス（取引番号=126,127）での分割後の企業（企業コード=”C5,6”）の取引はきちんと表現されているが，企業の分割という変化は表現されていないので, 分割前の企業の取引（取引番号=122,123）との関係は当然表現することはできない.

**セルラーDBMS での対応**

セルラーDBMS では企業分割の変化を表現することができる.

手順として，企業名＝”C5,6”の新しいインスタンス（obj-id_1=4,5）を追加し，そして企業が合併し新企業ができるという変化のホモトピーを保存する.

表 4.2.18　変更後の企業セル　$\mathcal{B}_c^4$

| obj-id_1 | **企業コード** | **企業名** | **本社所在地** |
|---|---|---|---|
| 1 | C1 | A | 東京 |
| 2 | C2 | B | 大阪 |
| 3 | C3 | C | 愛知 |
| **4** | **C5** | **E** | **東京** |
| **5** | **C6** | **F** | **神奈川** |

表 4.2.19　変化後の企業セル(オープン)　$e_c^1$

| obj-id_1 |
|---|
| 1 |
| 2 |
| 3 |
| **4** |
| **5** |

企業セル(オープン)において変化のホモトピーh1,h2 は

$$h1: 1 \to 4$$

$$h2: 1 \to 5$$

また, 取引記録として取引記録テーブルに新しいインスタンスを追加する.

表 4.2.20 取引記録セル　$\mathcal{B}_p^4$

| obj-id_2 | **取引番号** | **製品番号** | obj-id_1 |
|---|---|---|---|
| 1 | 122 | a | 1 |
| 2 | 123 | b | 1 |
| 3 | 124 | c | 2 |
| 4 | 125 | d | 3 |
| **5** | **126** | **a** | **4** |
| **6** | **127** | **b** | **5** |

このとき統合セルは以下になる。

表 4.2.21 統合セル　$\mathcal{B}_c^4 \sqcup \mathcal{B}_p^4 /\sim$

| obj -id_2 | **取引番号** | **製品番号** | obj -id_1 | **企業コード** | **企業名** | **本社所在地** |
|---|---|---|---|---|---|---|
| **1** | **122** | **a** | **1** | **C1** | **A** | **東京** |
| **2** | **123** | **b** | **1** | **C1** | **A** | **東京** |
| 3 | 124 | c | 2 | C2 | B | 大阪 |
| 4 | 125 | d | 3 | C3 | C | 愛知 |
| **5** | **126** | **a** | **4** | **C5** | **E** | **東京** |
| **6** | **127** | **b** | **5** | **C6** | **F** | **神奈川** |

この統合セルと保存されたホモトピーより, 分割前の企業（企業コード=”C1”）の取引（取引番号=122,123）と分割後の２企業（企業コード=”C5,6”）の取引（取引番号=126,127）の関係がきちんと表現されていることが分かる.

**事例 4）インスタンスの消滅・発生**

事象「ある時，１企業（企業コード＝”C3”）が取引対象企業の登録から抹消された．その後，再び取引企業としての再登録するになった．その後，その企業が製品番号=”a”の製品に対し取引を行った.」

**リレーショナル DBMS での対応**

リレーショナル DBMS では企業の登録抹消・再登録の変化の表現は不可能である.

手順として, 企業の登録抹消時, 一般的には企業コード=”C3”のインスタンスを削除する.

表 4.2.22　企業マスターテーブル　$R$

| **企業コード** | **企業名** | **本社所在地** |
|---|---|---|
| C1 | A | 東京 |
| C2 | B | 大阪 |

そして, 企業の再登録時, 一般的には新たに企業コード=”C7”のインスタンスを削除する.

表 4.2.23　企業マスターテーブル　$R$

| **企業コード** | **企業名** | **本社所在地** |
|---|---|---|
| C1 | A | 東京 |
| C2 | B | 大阪 |
| **C7** | **C** | **愛知** |

また, 取引記録として取引記録テーブルに新しいインスタンスを追加する.

表 4.2.24　取引記録テーブル $S$

| **取引番号** | **製品番号** | **企業コード** |
|---|---|---|
| 122 | a | C1 |
| 123 | b | C1 |
| 124 | c | C2 |
| 125 | d | C3 |
| **126** | **a** | **C7** |

このとき統合セルは以下になる。

表 4.2.25　JOIN 結合後のテーブル　$R \times S$

| **取引番号** | **製品番号** | **企業コード** | **企業名** | **本社所在地** |
|---|---|---|---|---|
| 122 | a | C1 | A | 東京 |
| 123 | b | C1 | A | 東京 |
| 124 | c | C2 | B | 大阪 |
| **126** | **a** | **C7** | **C** | **愛知** |

この結合後のテーブルでは, 追加されたインスタンス（取引番号=126）において, 最新情報である再登録後の企業の取引はきちんと表現されているが, 登録抹消前の前の取引については企業マスターにインスタンスがないために表現できない.

**セルラーDBMS での対応**

セルラーDBMS では企業の登録抹消時は null インスタンスの概念を導入することで表現することができる.

手順として, 登録抹消時は値が null であるインスタンス（obj-id_1=4）を追加し，その null インスタンスへの変化のホモトピーを保存する.

表 4.2.26　変更後の企業セル　$\mathcal{B}_c^4$

| obj-id_1 | **企業コード** | **企業名** | **本社所在地** |
|---|---|---|---|
| 1 | C1 | A | 東京 |
| 2 | C2 | B | 大阪 |
| **3** | **C3** | **C** | **愛知** |
| **4** | ***null*** | | |

企業セル(オープン)において変化のホモトピーh2 は

h1: 3 → 4

また，再登録時は企業コード=”C8”のインスタンスを追加する. そして null インスタンスから追加されたインスタンスへの変化のホモトピーを保存する.

表 4.2.27　変更後の企業セル　$\mathcal{B}_c^4$

| obj-id_1 | **企業コード** | **企業名** | **本社所在地** |
|---|---|---|---|
| 1 | C1 | A | 東京 |
| 2 | C2 | B | 大阪 |
| 3 | C3 | C | 愛知 |
| **4** | ***null*** | | |
| **5** | **C8** | **C** | **愛知** |

企業セル(オープン)において変化のホモトピーh2 は

h2: 4 → 5

また, 取引記録として取引記録テーブルに新しいインスタンスを追加する.

表 4.2.28 取引記録セル $\mathcal{B}_p^4$

| obj-id_2 | **取引番号** | **製品番号** | obj-id_1 |
|---|---|---|---|
| 1 | 122 | a | 1 |
| 2 | 123 | b | 1 |
| 3 | 124 | c | 2 |
| 4 | 125 | d | 3 |
| **5** | **126** | **a** | **5** |

このとき統合セルは以下になる。

表 4.2.29 統合セル　$\mathcal{B}_c^4 \sqcup \mathcal{B}_p^4 /\sim$

| obj -id_2 | **取引番号** | **製品番号** | obj -id_1 | **企業コード** | **企業名** | **本社所在地** |
|---|---|---|---|---|---|---|
| 1 | 122 | a | 1 | C1 | A | 東京 |
| 2 | 123 | b | 1 | C1 | A | 東京 |
| 3 | 124 | c | 2 | C2 | B | 大阪 |
| **4** | **125** | **d** | **3** | **C3** | **C** | **愛知** |
| **5** | **126** | **a** | **5** | **C8** | **C** | **愛知** |

この統合セルと保存されたホモトピーより, 登録抹消前の取引（取引番号=125）と再登録後の取引（取引番号=126）の関係がきちんと表現されていることが分かる.

**事例 5）インスタンスのキー属性の重複**

事象「ある時，企業 B（企業コード＝”C2”）の２つめの本社が兵庫に設けられ，本社所在地が大阪と兵庫になった．その後，その企業の兵庫本社が製品番号=”a”の製品に対し取引を行った.」

**リレーショナル DBMS での対応**

リレーショナル DBMS では，表 4.2.30 のように新たなインスタンス（企業コード=”C1”，企業名=”B”，本社所在地=”兵庫”）の追加は，主キー（企業コード）属性で値”C2”が重複するのでテーブル設計上不可能である.

表 4.2.30　企業マスターテーブル　$R$

| **企業コード** | **企業名** | **本社所在地** |
|---|---|---|
| C1 | A | 東京 |
| **C2** | **B** | **大阪** |
| **C2** | **B** | **兵庫** |
| C3 | C | 愛知 |

手順として，一般的には属性を本社所在地 1,2 に変更するなどデータベース管理者がテーブルの設計を変更する必要がある. こうなればデータベース設計全体を見直すことになり非常に手間がかかる.

表 4.2.31　企業マスターテーブル　$R$

| **企業コード** | **企業名** | **本社所在地** 1 | **本社所在地** 2 |
|---|---|---|---|
| C1 | A | 東京 | - |
| **C2** | **B** | **大阪** | **兵庫** |
| C3 | C | 愛知 | - |

**セルラー**DBMS **での対応**

セルラーDBMS では, 自動付番するオープンセル属性 obj-id_1 があるので，新しい本社に対するインスタンスを追加してもキー属性の値は重複しない. またインスタンス間の同値関係の対応が可能なので企業 B に関する既存のインスタンスとも関連を持つことができる.

手順として，まず新たに２つのインスタンス（obj-id_1=”4”，企業コード=”C2”，企業名=”B”，本社所在地=”大阪”）（obj-id_1=”5”，企業コード=”C2”，企業名=”B”，本社所在地=”兵庫”）を追加し、インスタンス間のホモトピーの変化と同値関係の対応をとる.

表 4.2.32　変更後の企業セル　$\mathcal{B}_c^4$

| obj-id_1 | **企業コード** | **企業名** | **本社所在地** |
|---|---|---|---|
| 1 | C1 | A | 東京 |
| 2 | C2 | B | 大阪 |
| 3 | C3 | C | 愛知 |
| **4** | **C2** | **B** | **大阪** |
| **5** | **C2** | **B** | **兵庫** |

そして，企業セル(オープン)において変化のホモトピーh4 は

h: 2 → 4, 5

企業セルの分解写像　c:

$\mathcal{B}_c^4 \to \mathcal{B}_c^2$（企業コード，企業名）$\sqcup_g \mathcal{B}_c^2$

アイデンティフィケーション関数 g:

$\partial\mathcal{B}_c^4 \to \mathcal{B}_c^2$（企業コード，企業名）

インスタンス間の同値関係の対応　f, g:

f: $\partial$ x (C2, B) → $\partial$ y (C2, B) / ~

g: $\partial$ x (C2, B) → $\partial$ z (C2, B) / ~

また, 取引記録として取引記録テーブルに新しいインスタンスを追加する.

表 4.2.33 取引記録セル $\mathcal{B}_p^4$

| obj-id_2 | **取引番号** | **製品番号** | obj-id_1 |
|---|---|---|---|
| 1 | 122 | a | 1 |
| 2 | 123 | b | 1 |
| 3 | 124 | c | 2 |
| 4 | 125 | d | 3 |
| **5** | **126** | **a** | **5** |

このとき統合セルは以下になる。

表 4.2.34 統合セル　$\mathcal{B}_c^4 \sqcup \mathcal{B}_p^4 /\sim$

| obj -id_2 | **取引番号** | **製品番号** | obj -id_1 | **企業コード** | **企業名** | **本社所在地** |
|---|---|---|---|---|---|---|
| 1 | 122 | a | 1 | C1 | A | 東京 |
| 2 | 123 | b | 1 | C1 | A | 東京 |
| **3** | **124** | **c** | **2** | **C2** | **B** | **大阪** |
| 4 | 125 | d | 3 | C3 | C | 愛知 |
| **5** | **126** | **a** | **5** | **C2** | **B** | **兵庫** |

この統合セルと保存されたホモトピーと同値関係の

対応より，兵庫本社設立前の取引（取引番号=124）と設立後の取引（取引番号=126）の関係がきちんと表現されていることが分かる．

**事例 6）1 つの実体に対するインスタンスの重複**

事象「ある時，１つの企業 A に対して２つの記録（企業コード＝”C1,4”）が重複して存在していたことが分かったので，改めて統一して管理を行う．その後，企業 A が製品番号=”a”の製品に対し取引を行った．」

**リレーショナル DBMS での対応**

リレーショナル DBMS では，一般的に記録の重複が分かったときにはどちらかの記録を削除し，関連する全テーブルの全データの修正を行う．よって非常に手間がかかる．

手順として，片方のインスタンス（企業コード=”C4”）を削除し，関連する取引記録テーブルのデータの修正を行う．

表 4.2.35　企業マスターテーブル　*R*

| 企業コード | 企業名 | 本社所在地 |
|---|---|---|
| **C1** | **A** | **東京** |
| C2 | B | 大阪 |
| C3 | C | 愛知 |
| **C4** | **A** | **東京** |

表 4.2.36　変更後の企業マスターテーブル　*R*

| 企業コード | 企業名 | 本社所在地 |
|---|---|---|
| C2 | B | 大阪 |
| C3 | C | 愛知 |
| **C4** | **A** | **東京** |

表 4.2.37　取引記録テーブル *S*

| 取引番号 | 製品番号 | 企業コード |
|---|---|---|
| 122 | a | **C1** |
| 123 | b | **C1** |
| 124 | c | C2 |
| 125 | d | C3 |

表 4.2.38　変更後の取引記録テーブル *S*

| 取引番号 | 製品番号 | 企業コード |
|---|---|---|
| 122 | a | **C4** |
| 123 | b | **C4** |
| 124 | c | C2 |
| 125 | d | C3 |

**セルラー**DBMS **での対応**

セルラーDBMS では，インスタンス間の同値関係の対応が可能なので関連を表現することができる．

手順として，重複するインスタンス間で同値関係の対応をとる．

表 4.2.39　変更後の企業セル　$\mathcal{B}_c^4$

| obj-id_1 | 企業コード | 企業名 | 本社所在地 |
|---|---|---|---|
| **1** | **C1** | **A** | **東京** |
| 2 | C2 | B | 大阪 |
| 3 | C3 | C | 愛知 |
| **4** | **C4** | **A** | **東京** |

インスタンス間の同値関係の対応　f:

f:　x (1, C1, A, 東京) → y (4, C4, A, 東京) / ~

また，取引記録として取引記録テーブルに新しいインスタンスを追加する．

表 4.1.40 取引記録セル $\mathcal{B}_p^4$

| obj-id_2 | 取引番号 | 製品番号 | obj-id_1 |
|---|---|---|---|
| 1 | 122 | a | 1 |
| 2 | 123 | b | 1 |
| 3 | 124 | c | 2 |
| 4 | 125 | d | 3 |
| **5** | **126** | **a** | **4** |

このとき統合セルは以下になる。

表 4.2.41 統合セル　$\mathcal{B}_c^4 \sqcup \mathcal{B}_p^4$ /~

| obj -id_2 | 取引番号 | 製品番号 | obj -id_1 | 企業コード | 企業名 | 本社所在地 |
|---|---|---|---|---|---|---|
| **1** | **122** | **a** | **1** | **C1** | **A** | **東京** |
| **2** | **123** | **b** | **1** | **C1** | **A** | **東京** |
| 3 | 124 | c | 2 | C2 | B | 大阪 |
| 4 | 125 | d | 3 | C3 | C | 愛知 |
| **5** | **126** | **a** | **4** | **C4** | **A** | **東京** |

この統合セルと同値関係の対応より，企業 A の２つのインスタンスが同じ実体としてきちんと表現されていることが分かる．

## 5.　結論

既存の汎用データベースであるリレーショナル DBMS はワールドモデルとしてデータの依存関係全体を把握しているデータベース管理者の存在を前提にしており，テーブルのデータとその依存関係が動的に変化する状況へのシステム適用は非常に困難である．本論文では，リレーショナル DBMS 運用時にマスターテーブルがもたらす問題点を，システム運用現場へのヒアリング調査によって抽出し以下の 7 つのパターンに分類した．

1. あるインスタンスが他のインスタンスに変化したがあったときの対応が難しい．
2. 複数のインスタンスが結合して，新たなインスタンスが生成されたときの対応が難しい．
3. あるインスタンスが分解して，複数のインスタンスが生成されたときの対応が難しい．
4. あるインスタンスが発生したときの対応が難しい．
5. あるインスタンスが消滅したときの対応が難しい．
6. マスターテーブルのキー属性のデータ重複への対応が難しい．
7. 同じ実体に対して異なるインスタンスを作成したときの対応が難しい．

その原因として，以下の３点が考察された．

- インスタンス変化のホモトピーが保存できないこと
- インスタンス間の同値関係による対応がとれないこと
- null インスタンスの概念がないこと

そこで，優れた情報空間構築モデルであるセルラー

モデルを採用したセルラーDBMS を開発した. そして企業取引のシチュエーションを取り上げリレーショナル DBMS と比較しながら，セルラーDBMS によって上記の７つの問題点は全て解決可能であることを，事例により具体的に示した.

## 参考文献


[1] C. J. Date, “The Database Relational Model: A Retrospective Review and Analysis”, Addison Wesley Publishing Company, 2000.5.

[2] T.L.Kunii and H. S. Kunii, “A Cellular Model for Information Systems on the Web -Integrating Local and Global Information-”, Proceedings of 1999 International Symposium on Database Applications in Non-Traditional Environments (DANTE’99), November 28-30, 1999, Heian Shrine, Kyoto, Japan, Organized by Research Project on Advanced Databases, in cooperation with Information Proceeding Society of Japan, ACM Japan, ACM SIGMOD Japan, pp. 19-24, IEEE Computer Society Press, Los Alamitos, California, U.S.A.

[3] T. L. Kunii, “Creating a New World ins-ide Computers -Methods and Implications-”, Proc. of the Seventh Annual Conference of the Australian Society for Computers in Learning in Tertiary Education (ASCILITE 89), G. Bishop and J. Baker (eds.), pp. 28-51, Gold Coast, Australia, December 11-13, 1989, [also available as Technical Report 89-034, Dept. of Information Science, The University of Tokyo].

[4] T.L.Kunii, “Homotopy Modeling as World Modeling”, Proceedings of Computer Graphics International '99 (CGI99), (June 7-11, 1999, Canmore, Alberta, Canada) pp. 130-141, IEEE Computer Society Press, Los Alamitos, California, U. S. A.

[5] T.L.Kunii, “Valid Computational Shape Modeling: Design and Implementation”, International Journal of Shape Modeling, World Scientific, December 1999.

[6] Setrag Khoshafian, “Object-Oriented Databases” pp. 132-142, John Wiley & Son, 1993.

[7] E. F. Codd, “A Relational Model for Large Shared Data Banks,” Communications of the ACM, Vol. 13, No. 6, pp.377-387, June 1970.



児玉　敏男（正会員）

1997 年東京大学工学部卒業. 1999 年同大学院修士課程修了．同年前田建設工業（株）入社，現在に至る．2001 年法政大学大学院 ITPC 修了．主にセルモデルを用いた DBMS の開発に従事.



國井　利泰（フェロー会員）

1962 年東京大学理学部卒業. 1964 年同大学院修士課程修了. 1967 年同大学院博士課程修了，理学博士．1978～93 年同大学理学部情報科学科教授．1993～1997 年会津大学学長兼教授．1998 年～法政大学教授，現在に至る．2000 年東京大学，会津大学名誉教授，現在に至る．主として，サイバーワールドに関する研究とその若手研究者の教育に従事．1998 年に IEEE CS より Taylor L. Booth Education Award を受賞．IEEE，IPSJ 各フェロー.