

### 記事

[Toshihiko Minamoto](#) · 2021年2月25日 9m read

## グローバルをクラスにマッピングする技術 (3/3)

古くなった MUMPS アプリケーションに新たな生命を吹き込みたいとお考えですか？

以下にご紹介するステップを実行すれば、グローバルをクラスにマッピングし、美しいデータを Object や SQL に公開できます。

上の内容に馴染みが無い方は、以下の記事を初めからお読みください。

[グローバルをマッピングする技術 1](#)

[グローバルをマッピングする技術 2](#)

この記事の例では、典型的な親子構造をマッピングする方法をお見せします。

前回と同じ免責事項: これらの記事を読んでもグローバルがよく理解できないという方は、WRC ([Support@InterSystems.com](mailto:Support@InterSystems.com)) までメールでお問い合わせください。喜んでサポートさせていただきます。

グローバルをクラスにマッピングするステップ。

1. グローバルデータが繰り返し使用されるパターンを特定する。
2. 固有キーの構成を特定する。
3. プロパティとそれぞれの型を特定する。
4. クラス内のプロパティを定義する (変数の添え字をお忘れなく)。
5. IdKey のインデックスを定義する。
6. Storage Definition を以下の手順で定義する。
  - a. 添え字を IdKey まで (IdKey を含む) 定義する。
  - b. Data セクションを定義する。
  - c. Row ID セクションには触れない。デフォルトが 99% の割合で適切なので、これはシステムに任せます。
7. クラス / テーブルをコンパイルし、テストします。

前回の例では、家族のメンバーは Activity をそれぞれ 1 つずつしか持てませんでした。今回は複数の Activity を持てるようにしたいと思います。以下の例のステップ 1 は、繰り返し使用されるデータが 2 種類あるので、少しだけ複雑になります。

```
^ParentChild(1)="Brendan^45956"
```

```
^ParentChild(1,"Hobbies",1)="Pit Crew"
```

```
^ParentChild(1,"Hobbies",2)="Kayaking"
```

```
^ParentChild(1,"Hobbies",3)="Skiing"
```

```
^ParentChild(2)="Sharon^46647"
```

```
^ParentChild(2,"Hobbies",1)="Yoga"
```

^ParentChild(2,"Hobbies",2)="Scrap booking"

^ParentChild(3)="Kaitlin^56009"

^ParentChild(3,"Hobbies",1)="Lighting Design"

^ParentChild(3,"Hobbies",2)="pets"

^ParentChild(4)="Melissa^56894"

^ParentChild(4,"Hobbies",1)="Marching Band"

^ParentChild(4,"Hobbies",2)="Pep Band"

^ParentChild(4,"Hobbies",3)="Concert Band"

^ParentChild(5)="Robin^57079"

^ParentChild(5,"Hobbies",1)="Baking"

^ParentChild(5,"Hobbies",2)="Reading"

^ParentChild(6)="Kieran^58210"

^ParentChild(6,"Hobbies",1)="SUBA"

^ParentChild(6,"Hobbies",2)="Marching Band"

^ParentChild(6,"Hobbies",3)="Rock Climbing"

^ParentChild(6,"Hobbies",4)="Ice Climbing"

ステップ 1:

このグローバルには、繰り返し使用されるデータが 2 種類あります。1 つ目のデータは、添え字の最初のレベルにあり、個人情報が格納されています。

^ParentChild(1)="Brendan^45956"

^ParentChild(2)="Sharon^46647"

^ParentChild(3)="Kaitlin^56009"

^ParentChild(4)="Melissa^56894"

^ParentChild(5)="Robin^57079"

^ParentChild(6)="Kieran^58210"

2 つ目のデータは、添え字の 3 つ目のレベルにあり、趣味が格納されています。

^ParentChild(1,"Hobbies",1)="Pit Crew"

^ParentChild(1,"Hobbies",2)="Kayaking"

^ParentChild(1,"Hobbies",3)="Skiing"

```
^ParentChild(2,"Hobbies",1)="Yoga"
```

```
^ParentChild(2,"Hobbies",2)="Scrap booking"
```

...

クラスは合計 2 つ、繰り返し使用されるデータのブロックそれぞれに 1 つずつ定義します。

ステップ 2:

クラスが 2 つあるということは、固有の識別子が各テーブル (Example3Parent と Example3Child) に 1 つずつ、合計で 2 つ必要になるということです。Parent テーブルは添え字が 1 つしかないのが簡単です。一方の Child テーブルには、最初の添え字 (Parent への参照) と 3 つ目の添え字 (childsub) を組み合わせた複合キーが使用されます。

ステップ 3:

データを見ていると、簡単に 5 種類のプロパティを特定できます。Example3Parent には、ParentId、Name、DateOfBirth と、プロパティが 3 つあり、Example3Child には 2 つ、ChildId と Hobby があります。この例では、プロパティをあと 2 つ定義する必要があります。これらは、Relationship Properties (関係プロパティ) と呼ばれています。別のクラスとの関係を定義するもので、各クラスに 1 つずつ記述します。親クラスは子クラスを多く持てる一方で、子クラスは親クラスを 1 つしか持てません。

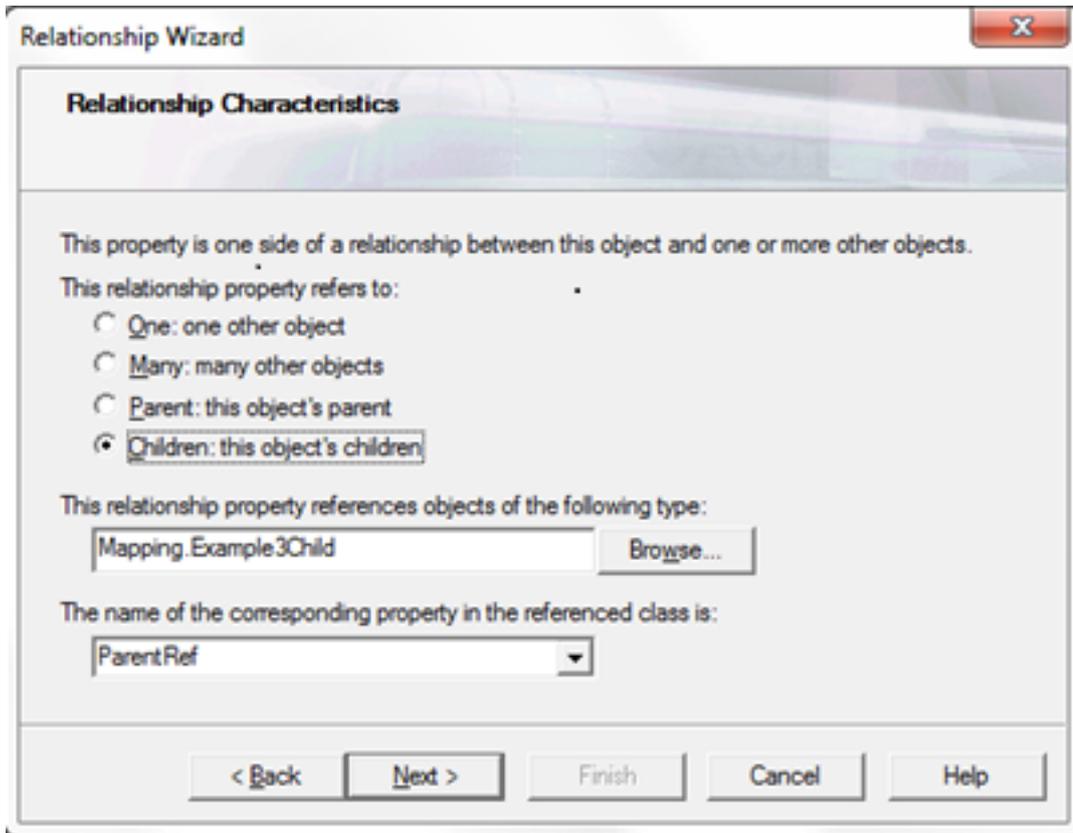
親クラスには、以下が定義されています。

```
Relationship HobbyRef As Mapping.Example3Child [ Cardinality = children, Inverse = ParentRef ];
```

子クラスでは、以下が定義されています。

```
Relationship ParentRef As Mapping.Example3Parent [ Cardinality = parent, Inverse = HobbyRef ];
```

これらのプロパティは、Property ウィザードを使えば定義できます。Relationship オプションボタンをクリックすると、以下のページが表示されます。



ステップ 4:

ステップ 3 で少し先走って Relationship プロパティを紹介しましたが、それ以外のプロパティは以下のとおりです。

Example3Parent:

Property Name As %String;

Property DateOfBirth As %Date;

Property ParentId As %Integer;

Example3Child:

Property Hobby As %String;

Property ChildId As %Integer;

ステップ 5:

Example3Parent の IdKey はいたって単純。添え字のレベルは 1 つだけです。

Index Master On ParentId [ IdKey ];

一方の Example3Child の IdKey は、添え字 1 と 3 に基づいています。しかし、親子関係は少しややこしくなります (十代の女の子をお持ちの親御さんなら共感していただけるのではないかと思います)。 IdKey インデックスには、ParentRef と ChildId の両方、または ChildId だけを指定することができますが、どちらの場合でもきちんとコンパイルします。

Index Master On (ParentRef, ChildId) [ IdKey ];

もしくは

Index Master On ChildId [ IdKey ];

個人的には最初の方が適切だと思いますが、私は面倒くさがり屋なので、自分がお見せする例には2つ目の例が登場する可能性が高いです。

ステップ 6:

Example3Parent については何の特記事項もありませんので、細かい説明は割愛します。

一方の Example3Child では、添え字の扱いが少し違うので、その手順を説明します。

ステップ 6a:

下にあるのは、子クラスのグローバルです。

```
^ParentChild({ParentRef},"Hobbies",{ChildId})= " "
```

添え字は3つ必要なことが分かります。1つ目の添え字に手こずる人が多いです。先ほどお見せした Relationship プロパティは使わず、代わりに親クラスのプロパティを使います。

それを行うには、完全なフィールド構文 {SchemaName.TableName.FieldName} を使います。今回の例だと、1つ目の添え字は {Mapping.Example3Parent.ParentId} となり、

2つ目の添え字は定数の “ Hobbies ”、

そして3つ目にはフィールド {ChildId} を使います。

ステップ 6b:

Data セクションは極めてシンプル。フィールド {Hobby} だけです。Piece も Delimiter もありません。

ステップ 6c:

では皆さんと一緒に「お見せするものがないので、空白にしておきます！」(笑)

ステップ 7:

後はコンパイルするだけです。

```
Compilation started on 11/28/2016 08:26:31 with qualifiers 'fuk/importselectivity=1 /checkuptodate=expandedonly'  
Compiling 2 classes, using 2 worker jobs  
Compiling class Mapping.Example3Parent  
Compiling class Mapping.Example3Child  
Compiling table Mapping.Example3Parent  
Compiling table Mapping.Example3Child
```

```
Compiling routine Mapping.Example3Parent.1
Compiling routine Mapping.Example3Child.1
Compilation finished successfully in 0.900s.
```

そして、単純な JOIN を実行し、データが正しく表示されるのを確認します。

```
SELECT P.ParentId, P.Name, P.DateOfBirth, C.ID, C.Hobby
FROM Mapping.Example3Parent P
JOIN Mapping.Example3Child C ON P.ParentId = C.ParentRef
```

ParentId	Name	DateOfBirth	ID	Hobby
1	Brendan	10/28/1966	1  1	Pit Crew
1	Brendan	10/28/1966	1  2	Kayaking
1	Brendan	10/28/1966	1  3	Skiing
2	Sharon	09/18/1968	2  1	Yoga
2	Sharon	09/18/1968	2  2	Scrap booking
3	Kaitlin	05/07/1994	3  1	Lighting Design
3	Kaitlin	05/07/1994	3  2	pets
4	Melissa	10/08/1996	4  1	Marching Band
4	Melissa	10/08/1996	4  2	Pep Band
4	Melissa	10/08/1996	4  3	Concert Band
5	Robin	04/11/1997	5  1	Baking
5	Robin	04/11/1997	5  2	Reading
6	Kieran	05/16/2000	6  1	SUBA
6	Kieran	05/16/2000	6  2	Marching Band
6	Kieran	05/16/2000	6  3	Rock Climbing
6	Kieran	05/16/2000	6  4	Ice Climbing

子クラスに ParentId\_ || ChildId で構成される ID というフィールドがあることに注目してください (どのクラスにも ID というフィールド / プロパティがあります)。

自分で入力したくないという方は、グローバルとクラスが記述されたこちらのファイルをお使いください:  
[MappingExample3.zip](#)

[#マッピング](#) [#SQL](#) [#グローバル](#) [#Caché](#)

---

### ソースURL:

<https://jp.community.intersystems.com/post/%E3%82%B0%E3%83%AD%E3%83%BC%E3%83%90%E3%83%AB%E3%82%92%E3%82%AF%E3%83%A9%E3%82%B9%E3%81%AB%E3%83%9E%E3%83%83%E3%83%94%E3%83%B3%E3%82%B0%E3%81%99%E3%82%8B%E6%8A%80%E8%A1%93-33>