The relational underpinnings of CIF

James Hester

ANSTO, Sydney
How CIF transfers meaning

CIF needs humans:

```
_definition.id 'pd_proc.intensity_total'
_description.text

Intensity values for the processed diffractogram at each data point where background, normalization, or other corrections have not been applied.

_name.category_id pd_proc
_name.object_id intensity_total
_type.contents Real
_enumeration.range 0.0:
```

Whitespace-separated columns of ASCII-formatted numbers. Comments on lines with first character “#”.

```
_pd_proc.intensity_total is in column 2.
```

# Data from file WBT0069064.nx.hdf

<table>
<thead>
<tr>
<th>Angle</th>
<th>Intensity</th>
<th>Error</th>
</tr>
</thead>
<tbody>
<tr>
<td>15.08374</td>
<td>3297.29</td>
<td>96.79420</td>
</tr>
<tr>
<td>15.14624</td>
<td>2749.89</td>
<td>34.75392</td>
</tr>
<tr>
<td>15.20874</td>
<td>3046.08</td>
<td>55.95003</td>
</tr>
<tr>
<td>15.27124</td>
<td>3043.61</td>
<td>50.83487</td>
</tr>
<tr>
<td>15.33374</td>
<td>3028.87</td>
<td>57.34422</td>
</tr>
<tr>
<td>15.39624</td>
<td>3036.24</td>
<td>54.20394</td>
</tr>
<tr>
<td>15.45874</td>
<td>3066.64</td>
<td>51.41039</td>
</tr>
<tr>
<td>15.52124</td>
<td>3058.82</td>
<td>62.81126</td>
</tr>
<tr>
<td>15.58374</td>
<td>2966.74</td>
<td>53.42696</td>
</tr>
<tr>
<td>15.64624</td>
<td>2900.90</td>
<td>54.75103</td>
</tr>
<tr>
<td>15.70874</td>
<td>2911.23</td>
<td>51.58636</td>
</tr>
<tr>
<td>15.77125</td>
<td>2852.62</td>
<td>50.36857</td>
</tr>
<tr>
<td>15.83375</td>
<td>2866.92</td>
<td>54.19454</td>
</tr>
<tr>
<td>15.89625</td>
<td>2852.62</td>
<td>50.36857</td>
</tr>
<tr>
<td>15.95875</td>
<td>2911.23</td>
<td>51.58636</td>
</tr>
</tbody>
</table>
```
Why machine-readable dictionaries?

- For validation
- For transformation to other formats
- For precision of description
- For automated database construction!

```
save_topol_atom.node_id

    _definition.id    '_topol_atom.node_id'
    _description.text

    ;
    The node associated with this atom, if applicable. It must match a value provided for _topol_node.id.

    ;
    _name.linked_item_id    '_topol_node.id'
    _type.purpose         Link
    _type.contents        Integer
    _enumeration.range    1:

    save_

save_topol_atom.symop_id

    _definition.id    '_topol_atom.symop_id'
    _description.text

    ;
    The identifier of the symmetry operation that is to be applied to the coordinates of the atom given by _topol_atom.atom_label before addition of the translations given by _topol_atom.translation. The value must match a value of _space_group_symop.id. If this item is omitted or assigned to '.', the identity operation is assumed.

    ;
    _name.linked_item_id    '_space_group_symop.id'
    _type.contents        Integer
    _enumeration.range    1:192
    _enumeration.default  1

    save_
```
1. Introduce relational model
2. Introduce category theory
3. Link the two together
4. CIF is relational
5. Why is this useful
6. Multiple data blocks
## The Relational Model

- A relation has no repeated column names, or row contents
- The order of columns and rows has no meaning
- Each column has an associated domain from which its values are drawn
- "Key" (or candidate key): values can be used to select unique row

### Table: "Relation" = Table

<table>
<thead>
<tr>
<th>label</th>
<th>x</th>
<th>y</th>
<th>z</th>
<th>element</th>
</tr>
</thead>
<tbody>
<tr>
<td>O1A</td>
<td>.5505</td>
<td>.6374</td>
<td>.1605</td>
<td>O</td>
</tr>
<tr>
<td>C1A</td>
<td>.4170</td>
<td>.6930</td>
<td>.4954</td>
<td>C</td>
</tr>
<tr>
<td>C2A</td>
<td>.3145</td>
<td>.6704</td>
<td>.6425</td>
<td>C</td>
</tr>
</tbody>
</table>

- **Key**: column(s) whose values always select unique row

- **"Attribute"**: Column Header
- **"Tuple"**: Row

### Notes:

- "Relation" = Table
- "Attribute" = Column Header
- "Tuple" = Row
Relations refer to particular rows in other relations by referencing the values of their key data names.
Each column in a relation is a function mapping the key data name(s) to the set from which the column values are drawn.

\[
x(O1A) = 0.5505
\]

\[
\text{Length}(C1A, O1A) = 1.364
\]

\[
\text{element}(O1A) = O
\]
"Normal forms": efficient relations

- Focused on improving database performance
- Reduce duplication of information
- Improve orthogonality of information
- Allow parallel operations on databases

We just want:

- To reduce information duplication
- Minimum disruption for future expansion: maximally non-committal

Can I update a single row in my loop and maintain consistency/truth?
Third normal form

- All non-key data names depend only on the values of the keys
- Key data names do not depend on one another
- No transitive dependencies allowed!
- "every non-key column must provide a fact about the key, the whole key, and nothing but the key" (William Kent)\(^1\).

<table>
<thead>
<tr>
<th>label</th>
<th>x</th>
<th>y</th>
<th>z</th>
<th>element</th>
<th>atomic number</th>
</tr>
</thead>
<tbody>
<tr>
<td>O1A</td>
<td>.5505</td>
<td>.6374</td>
<td>.1605</td>
<td>O</td>
<td>8</td>
</tr>
<tr>
<td>C1A</td>
<td>.4170</td>
<td>.6930</td>
<td>.4954</td>
<td>C</td>
<td>6</td>
</tr>
<tr>
<td>C2A</td>
<td>.3145</td>
<td>.6704</td>
<td>.6425</td>
<td>C</td>
<td>6</td>
</tr>
</tbody>
</table>

Example of transitive dependency

\(^{1}\)William Kent, "A Simple Guide to Five Normal Forms in Relational Database Theory", *Communications of the ACM* (1983)\(^{26}(2)\), 120-125
CIF dictionaries describe relations

Table (category) is given a name: **ATOM_SITE**

Key data name(s) identified using **_category_key.name**

Data names to be tabulated in this table assigned via **_name.category_id**

Data name that links to another category's data name given in **_name.linked_item_id**

<table>
<thead>
<tr>
<th>Atom Site Information</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>_name.category_id</strong></td>
</tr>
<tr>
<td><strong>_name.object_id</strong></td>
</tr>
<tr>
<td><strong>_category_key.name</strong></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Atom Site Details</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>_atom_site.label</strong></td>
</tr>
<tr>
<td><strong>_atom_site.fract_x</strong></td>
</tr>
<tr>
<td><strong>_atom_site.fract_y</strong></td>
</tr>
<tr>
<td><strong>_atom_site.fract_z</strong></td>
</tr>
<tr>
<td><strong>_atom_site.type_symbol</strong></td>
</tr>
<tr>
<td>o1</td>
</tr>
<tr>
<td>o2</td>
</tr>
<tr>
<td>c1</td>
</tr>
</tbody>
</table>
(Mathematical) Category theory

- Objects connected by arrows ("morphisms")
- Arrows can be composed \((f \circ g)\) is also a morphism)
- All objects have an identity morphism \((f \circ i = f)\)
- Many, many theorems.
Relations are categories over sets and functions

- Category theory "objects" → Sets
- Category theory "morphisms" → Functions
- A relation tabulates values for the functions
- Symbol "o": composition gets back to original value
- Green arrows: function to obtain atomic number from bond.

...CIF dictionaries describe categories!
- A CIF category ≈ category theory object
- A CIF data name ≈ morphism
Aside: Category theory is useful

Relations can be automatically transformed if corresponding objects are identified.


www.categoricaldata.net
1. (Functional picture) Given which key data names can I unambiguously determine a value for my data name?

2. Do I need a data name in this category or is there a path from the keys already? (composition of functions)

3. Is the dependence of a data name on the key via another data name in the same category? (remove transitivity)

4. There is not one perfect way to express the data relationships (functors) - choose a way that suits the context e.g. widespread convention
Our relational data sit inside containers (blocks, files, directories, ...)
There are no containers in the relational model
How do we combine container contents and retain the relational model?
Projection and Scoping

1. Collect rows that have a particular value of a chosen key data name (projection)

2. Drop columns for which the value can be inferred (scoping)
Reverting Projection and Scoping

1. Add back the missing columns (undo scoping)
2. Merge separate tables together (undo projection)
In CIF terms

- A "Set" category is one that has been projected and therefore takes single values for data names.
- Child data names of "Set" category key data names are elided.
- If key data names are provided for "Set" categories, information in that category from separate data blocks can be combined (multiple crystals, diffraction conditions, etc.).

```
_cat1.A 1
_cat1.Item1  20.34
_cat1.Item2  151.4

<table>
<thead>
<tr>
<th>_cat2.B</th>
<th>_cat2.Data</th>
</tr>
</thead>
<tbody>
<tr>
<td>x</td>
<td>1.1</td>
</tr>
<tr>
<td>y</td>
<td>3.2</td>
</tr>
<tr>
<td>z</td>
<td>2.2</td>
</tr>
<tr>
<td>q</td>
<td>4.1</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>_cat3.C</th>
<th>_cat3.Data</th>
</tr>
</thead>
<tbody>
<tr>
<td>RR</td>
<td>Perth</td>
</tr>
<tr>
<td>FF</td>
<td>Hobart</td>
</tr>
<tr>
<td>EE</td>
<td>Darwin</td>
</tr>
</tbody>
</table>
```

```
_cat1.A 2
_cat1.Item1  17.4
_cat1.Item2  132.1

<table>
<thead>
<tr>
<th>_cat2.B</th>
<th>_cat2.Data</th>
</tr>
</thead>
<tbody>
<tr>
<td>x</td>
<td>-1.1</td>
</tr>
<tr>
<td>y</td>
<td>1.0</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>_cat3.C</th>
<th>_cat3.Data</th>
</tr>
</thead>
<tbody>
<tr>
<td>EE</td>
<td>Perth</td>
</tr>
<tr>
<td>RR</td>
<td>Perth</td>
</tr>
<tr>
<td>SS</td>
<td>Perth</td>
</tr>
</tbody>
</table>
```

cat1 is a Set category
_cat2.A and _cat3.A are elided
_name.linked_item_id for _cat2.A and _cat3.A is _cat1.A
The key for cat1 is _cat1.A