**Introduction**

- Dimensions are not independent of time
- Dimension table attributes are relatively static but they are not fixed forever
- DW designers must proactively work with managers to determine the appropriate change-handling strategy
- For each dimension table attribute we must specify a strategy to handle changes

**Slowly Changing Dimension**

Tadeusz Morzy
based on
The Data Warehouse Toolkit

**SCD 0: retain original**

- With type 0, the dimension attribute value never changes
- Facts are always grouped by the original value
- Type 0 is appropriate for any attribute labeled “original” (e.g. customer original score)
- Persistent durable keys are always type 0 attribute

**SCD 1: overwrite**

- With type 1 we overwrite the old attribute value in the dimension row, replacing it with the current value
- No dimension or fact table keys are modified
- The fact table is untouched

<table>
<thead>
<tr>
<th>Product Key</th>
<th>SKU (NK)</th>
<th>Product Name</th>
<th>Department Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>12345</td>
<td>ABC922-Z</td>
<td>IntelliKidz</td>
<td>Education</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Product Key</th>
<th>SKU (NK)</th>
<th>Product Name</th>
<th>Department Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>12345</td>
<td>ABC922-Z</td>
<td>IntelliKidz</td>
<td>Strategy</td>
</tr>
</tbody>
</table>
SCD 1: overwrite

- A type 1 response is appropriate if the attribute change is insignificant correction
- A type 1 response is also appropriate if there is no value in keeping the old description
- Some BI applications can produce different results before versus after the type 1 attribute change
  - Any preexisting aggregations based on the department value need to be rebuilt
  - In case of OLAP cube and type 1 attribute is a hierarchical rollup attribute, like department, the cube likely needs to be reprocessed when the type 1 attribute changes

SCD 2: add new row

- DW/BI should correctly represent history
- A type 2 response is the predominant technique for supporting this requirement

<table>
<thead>
<tr>
<th>Prod. Key</th>
<th>SKU</th>
<th>Product Name</th>
<th>Dept. Name</th>
<th>Row effective date</th>
<th>Row Expiration date</th>
<th>Current row indicator</th>
</tr>
</thead>
<tbody>
<tr>
<td>12345</td>
<td>ABC922</td>
<td>IntelliKidz</td>
<td>Education</td>
<td>2012-01-01</td>
<td>9999-12-31</td>
<td>current</td>
</tr>
<tr>
<td>25984</td>
<td>ABC922</td>
<td>IntelliKidz</td>
<td>Strategy</td>
<td>2013-02-01</td>
<td>9999-12-31</td>
<td>current</td>
</tr>
</tbody>
</table>

- Dimension and fact table keys are modified but you don’t have the historical fact table rows to modify the product key – in fact the fact table is untouched
- Advantages:
  - allows us to accurately keep all historical information.
- Disadvantages:
  - This will cause the size of the table to grow fast. In cases where the number of rows for the table is very high to start with, storage and performance can become a concern.
  - This necessarily complicates the ETL process.
- Type 2 slowly changing dimension should be used when it is necessary for the data warehouse to track historical changes

SCD 3: add new attribute

- In Type 3, there will be two columns to indicate the particular attribute of interest, one indicating the original value, and one indicating the current value
- There will also be a column that indicates when the current value becomes active

<table>
<thead>
<tr>
<th>Product Key</th>
<th>SKU (NK)</th>
<th>Product Name</th>
<th>Department Name</th>
<th>Prior Department Name</th>
<th>Effective date</th>
</tr>
</thead>
<tbody>
<tr>
<td>12345</td>
<td>ABC922-Z</td>
<td>IntelliKidz</td>
<td>Education</td>
<td>Education</td>
<td>2013-02-01</td>
</tr>
</tbody>
</table>
SCD 3: add new attribute

- **Advantages:**
  - This does not increase the size of the table, since new information is updated.
  - This allows us to keep some part of history.
- **Disadvantages:**
  - Type 3 will not be able to keep all history where an attribute is changed more than once. For example, if Department later changes again its name to Marketing on December 15, 2013, the previous name will be lost.
- **Usage:**
  - Type 3 is rarely used in actual practice.
  - Type 3 should only be used when it is necessary for the data warehouse to track historical changes, and when such changes will only occur for a finite number of time.

SCD 4: add mini-dimension

- The Type 4 technique is used when a group of dimension attributes are split off into a separate mini-dimension.
- This approach is useful when dimension attribute values are relatively volatile.
- Type 4 – also called "using historical table"
- A separate historical table is used to track all dimension's attribute historical changes for each of the dimension.
- The 'main' dimension table keeps only the current data e.g. customer and customer_history tables.
**SCD 4: add mini-dimension**

<table>
<thead>
<tr>
<th>Demographics Key</th>
<th>Age Band</th>
<th>Purchase Frequency Score</th>
<th>Income Level</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>21-25</td>
<td>low</td>
<td>&lt;30 000</td>
</tr>
<tr>
<td>2</td>
<td>21-25</td>
<td>Medium</td>
<td>&lt;30 000</td>
</tr>
<tr>
<td>3</td>
<td>21-25</td>
<td>High</td>
<td>&lt;30 000</td>
</tr>
<tr>
<td>4</td>
<td>21-25</td>
<td>Low</td>
<td>30000-39 999</td>
</tr>
<tr>
<td>121</td>
<td>26-30</td>
<td>Low</td>
<td>&lt;30 000</td>
</tr>
<tr>
<td>122</td>
<td>26-30</td>
<td>Medium</td>
<td>&lt;30 000</td>
</tr>
</tbody>
</table>

- Frequently-used attributes in multi-million row dimension tables are also mini-dimension design candidates, even if they don’t change frequently.
- A surrogate key is assigned to each unique profile or combination of attribute values in the mini-dimension.
- The surrogate keys of both the base dimension and mini-dimension profile are captured as foreign keys in the fact table.

**SCD 5, 6, 7**

- The Type 5, 6, and 7 techniques are hybrids that combine the basics to support the common requirement to both accurately preserve historical attribute values, plus report historical facts according to current attribute values.
- The hybrid approaches provide more analytic flexibility, albeit with greater complexity.

**SCD 5: Mini-dimension and Type 1 Outrigger**

- The type 5 technique builds on the type 4 mini-dimension by embedding a “current profile” mini-dimension key in the base dimension that’s overwritten as a type 1 attribute.
- This approach, called type 5 because 4 + 1 equals 5, allows the currently-assigned mini-dimension attribute values to be accessed along with the base dimension’s others without linking through a fact table.
- Logically, we typically represent the base dimension and current mini-dimension profile outrigger as a single table in the presentation layer.
SCD 5: Mini-dimension and Type 1 Outrigger

- The outrigger attributes should have distinct column names, like “Current Income Level,” to differentiate them from attributes in the mini-dimension linked to the fact table.
- The ETL team must update/overwrite the type 1 mini-dimension reference whenever the current mini-dimension changes over time.
- If the outrigger approach does not deliver satisfactory query performance, then the mini-dimension attributes could be physically embedded (and updated) in the base dimension.

SCD 6: add Type 1 attribute to Type 2 Dimension

- Type 6 builds on the type 2 technique by also embedding current attributes in the dimension so that fact rows can be filtered or grouped by either the type 2 value in effect when the measurement occurred or the attribute’s current value.
- The type 6 moniker was suggested by an HP engineer in 2000 because it’s a type 2 row with a type 3 column that’s overwritten as a type 1; both $2 + 3 + 1$ and $2 \times 3 \times 1$ equal 6.
- With this approach, the current attributes are updated on all prior type 2 rows associated with a particular durable key, as illustrated by the following sample rows.
SCD 6: add Type 1 attribute to Type 2 Dimension

- Type 6 - combine approaches of types 1, 2, 3 (1+2+3=6)
- We have in dimension table such additional columns as:
  - current_type - for keeping current value of the attribute. All history records for given item of attribute have the same current value
  - historical_type - for keeping historical value of the attribute. All history records for given item of attribute could have different values
  - start_date - for keeping start date of 'effective date' of attribute's history
  - end_date - for keeping end date of 'effective date' of attribute's history
  - current_flag - for keeping information about the most recent record. In this method to capture attribute change we add a new record as in type 2
  - The current_type information is overwritten with the new one as in type 1. We store the history in a historical_column as in type 3

SCD 7: Dual Type 1 and Type 2 Dimension

- With type 7, the fact table contains dual foreign keys for a given dimension: a surrogate key linked to the dimension table where type 2 attributes are tracked, plus the dimension's durable supernatural key linked to the current row in the type 2 dimension to present current attribute values
- Type 7 delivers the same functionality as type 6, but it’s accomplished via dual keys instead of physically overwriting the current attributes with type 6.
- Like the other hybrid approaches, the current dimension attributes should be distinctively labeled to minimize confusion

Summary

Finally, here’s a figure that highlights the implications of each slowly changing dimension technique on the analysis of fact table performance metrics