# Mass data entry on empty cells in tables with many dimensions on the axes

## Symptom

For tables with many dimensions on the axis, a mass data entry (or copying facts from an excel sheet) onto a high number of empty cells can lead to performance and/or memory issues.

## Background

The runtime and memory consumption of a mass data entry depend on the size of the cartesian hull of the filters representing the empty cells among the cells that are changed. In other words, they depend on the product of the number of unique members for all dimensions on the axes for the area of touched empty cells.

## Example

In the table shown below, data is supposed to be entered onto all cells for February & March. The cartesian hull for this area contains 158400 member tuples combinations (product of the numbers of unique members per dimension: 1 for Region, 11 for Department, 20 for Employee ID, 18 for Hire Date, 20 for Position and 2 for Date). While this mass data entry can be executed without problems, a similar mass data entry onto 100 rows instead of 20 might span a cartesian hull of billions of member combinations and lead to performance issues or fail due to too high memory consumption.



## Mitigation options

1. Link dimensions via validation rules.
2. Prefill cells that are supposed to be changed with zeros using a script data action.
3. Add input controls on dimensions to split up the mass data entry into multiple smaller parts.
4. Remove unnecessary dimensions from the axes.
5. Use data import jobs instead of copy-pasting from excel files.

## Option 1: Link dimensions via validation rules

If some dimensions are semantically related to each other, these dimensions should be linked via **validation rules** (see below). **This will reduce the complexity and runtime significantly**, as only tuples with valid member combinations will be processed.

Example

In the given example, Hire Date and Position are properties of the Employee ID dimension rather than semantically separate dimensions. In fact, the Employee ID dimension has properties with the same names:



Therefore, the complexity can be reduced by creating a validation rule linking the properties “Hire Date” and “Position” of the Employee ID dimension with the dimensions of the respective name:



## Option 2: Prefill with zeros

If none of the dimensions on the axes are semantically related, and no other ways to reduce complexity are suitable either, one can circumvent performance and memory issues by prefilling the area that is to be changed with zeros using a data action. (Mass data entries on booked cells are faster and require less memory, as they require less complex calculations.)
**Important:** Only prefill the area edited by the mass data entry, as a high number of unnecessary zero value records can impact performance (<https://launchpad.support.sap.com/#/notes/3102951>).

### Example

The following data action script step prefills empty cells for measure Salary in the data area given by Region Ireland and Date February & March 2023 for tuples where data exists for January 2023 without affecting already existing records in said area:

MEMBERSET [d/Measures] = "Salary"

MEMBERSET [d/Region] = "Ireland"

MEMBERSET [d/Date] = ("202302", "202303")

DATA.APPEND() = RESULTLOOKUP([d/Date] = "202301") \* 0





## Option 3: Add input controls on dimensions to split up the mass data entry into multiple smaller mass data entries

A further option to reduce complexity is to add input controls on one or more of the dimensions on the axes and split up the mass data entry into one per dimension member. Splitting up the mass data entry in such a way can decrease the variation along other dimensions too; as an effect, several mass data entries onto smaller data regions can be faster as one mass data entry onto the whole area.

Example

In the given scenario, one can add an input control on the Department dimension and enter the values for each department separately:



## Option 4: Remove unnecessary dimensions from the axes

Removing unneeded dimensions from the axes can also help improving performance.

Example

The dimension “Hire Date” might be regarded as unnecessary for the table with the salary information. (One can use validation rules to directly assign to the fitting members for this dimension, or use an allocation process or a data action to distribute along this dimension later on.)



## Option 5: Use data import jobs instead of copy-pasting from excel files

Instead of copy-pasting large amounts of data from an excel file into a table, it might be advised to use a data import job instead. For more information, see <https://help.sap.com/docs/SAP_ANALYTICS_CLOUD/00f68c2e08b941f081002fd3691d86a7/a8435da6970041d2beb3299cdcff7026.html>.