



Further playing with SAP HANA News on performance-related matters in SPS 7

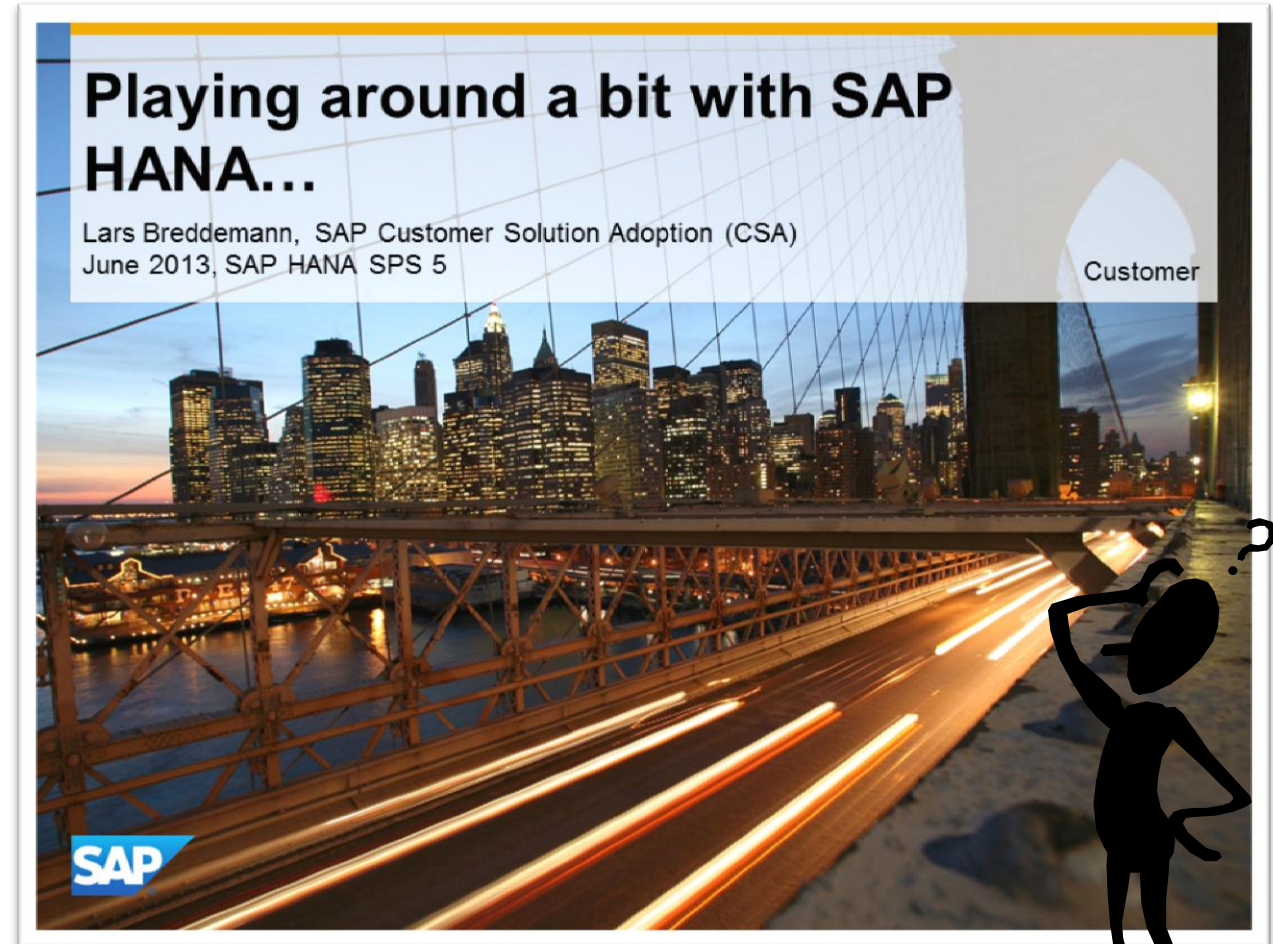


Lars Breddemann, Strategic Customer Engagement (SCE), P&I

MultiColumnJoins

SAP HANA SPS 5 learnings recap

- Special hidden columns - CONCAT_ATTRIBUTE - are created and maintained in the background
- Visible as `$col1$col2$...$` in `m_cs_all_columns`
- These hidden columns can take a lot of memory
- No officially documented way to drop these hidden columns (although it's possible)
- Multi-column Joins require these columns
- Same for multi-column primary keys
- secondary indexes can be created on column store tables and that can actually be a good idea



<http://scn.sap.com/docs/DOC-46329>

Joins via concats recap

- First (ever) execution of join or activation of information model slow due to build up of concat attribute
- concats typically cannot be compressed well + dictionary size large due to concatenated values
- DELTA MERGE takes more time and more memory
- concats can not be used if only some of the columns are used in the join → a new concat is created for each set of join columns
- concat are created on **both** joined tables
- Joins via concats use the single column join code → same speed as single
- Special case: primary key constraint → hidden column \$trexexternalkey\$
- Dropping possible via
`ALTER TABLE <table_name>`
`WITH PARAMETERS ( 'DELETE_CONCAT_ATTRIBUTE' = '<concat_attribute_name>' )`

SPS 7

MultiColumnJoin (MCJ) w/o concat_attribute possible

- Joins now can work on multiple columns
- No permanent space usage required any more
- During join processing more data needs to be processed ad hoc, thus multi column joins are slower without concat_attribute

- Usage can be set via DB parameters:

`indexserver.ini` → `[JOINS]` → `OPTIMIZE_METAMODEL = ON | OFF | AUTO (Default)`

ON: create + use concats, **OFF:** don't create or use concats, **AUTO:** use but don't create concats

- Information models & SAP BW still create the concat attributes for better performance
- **Attention:** with rev. 70/71 still problems, SAP note 1979367: “As of SAP HANA Revision 72, the “MultiColumnJoin” is fully supported”

{DEMO}

What are column store indexes?

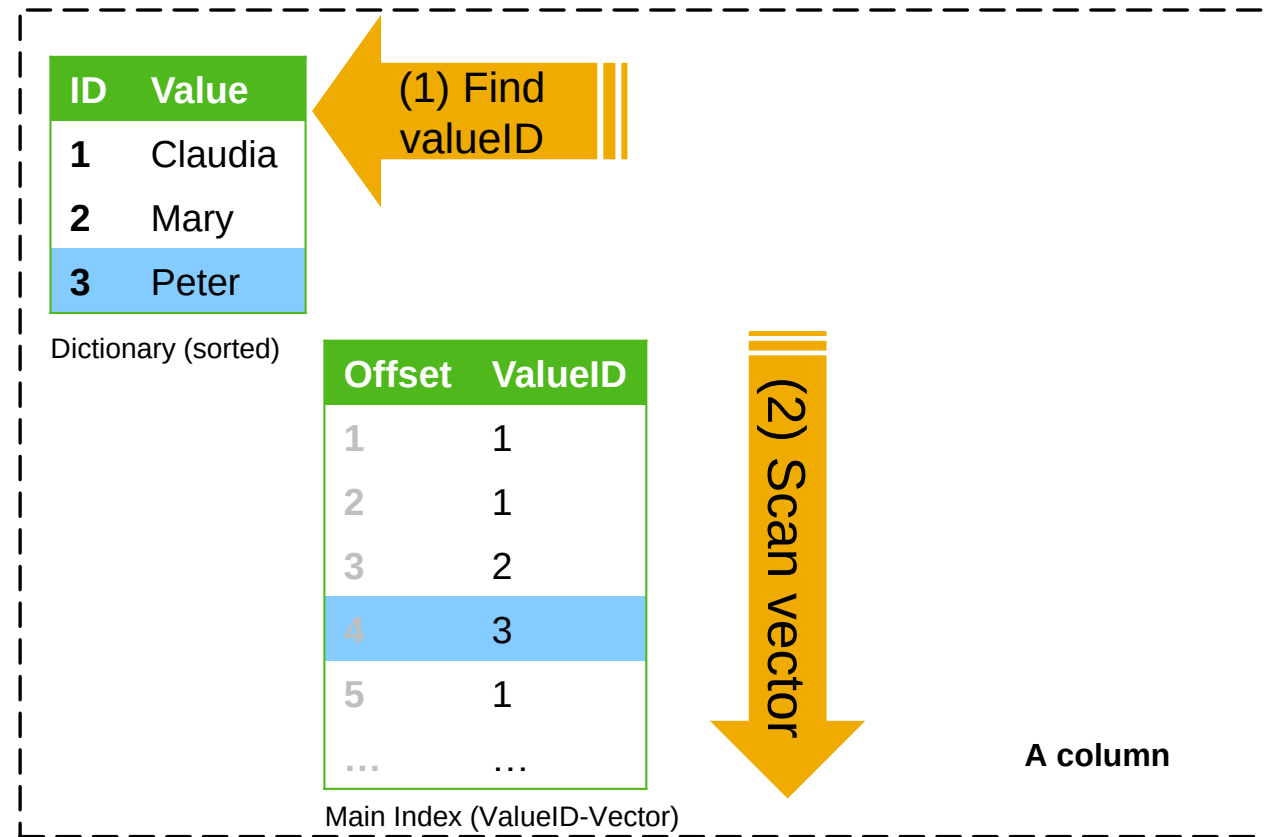
Column store review

Column (in MAIN store) consists of a sorted dictionary and a main index/vector

To find records with a specific value, the dictionary is read via binary search to find the valueID.

Then the whole main vector is scanned to find all occurrences of the valueID.

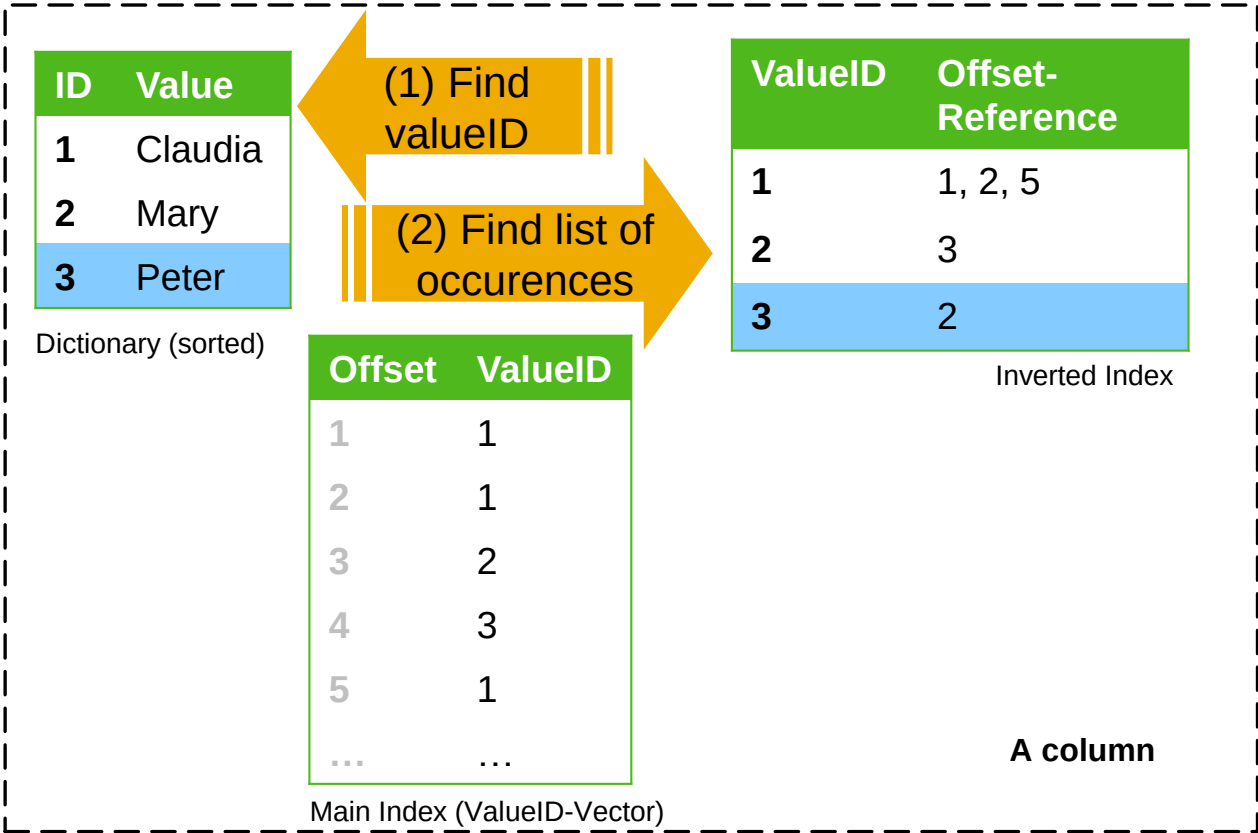
Even if we know that the value can only be there once, the whole vector must be scanned.



Column store – inverted indexes

To avoid the scan, the ‘inverted index’ represents the same data in a different way: all occurrences of a specific valueID are stored in a list next to the valueID.

That way the time to lookup only depends on the number of occurrences and not on the total number of records.



Column store – inverted indexes

Note:

- Columns in the column store are separate, independent data and complex data structures.
- The inverted index is an **optional** part of the main store of a column store column.
- It lives in the main-part of a column store table.

(as the delta part of a column store table is organized a lot different, this whole discussion only is about main store indexes!)

Creating inverted indexes

Inverted indexes can be created via several ways:

- Create index DDL command
- Single column primary key definition
- Multi column primary key definition
- Single column unique constraint
- Multi column unique constraint
- Implicit concatenated column creation

Differences to classic indexes

- **No Index-only strategy possible to avoid touching the table**
It just doesn't make any sense, as the inverted indexes is part of the column data structure.
- ~~**Inverted indexes are re-created upon loading time of the column**~~
~~This means there is an additional delay for the first usage after loading.~~
- **As of SPS 7 inverted indexes are also persisted to disk and usable after loading**
(for rev. 70 mind SAP note 1976994 - Wrong data results on persisted indexes)
- **Inv. indexes doesn't speed up min/max queries.**
These can be answered via the column dictionary
- **Inv. indexes don't help with reading data in a sorted way.**
Remember, the data from the delta log must be read as well and the index exists in the main-part only. Thus, a separate sorting step is required anyhow.

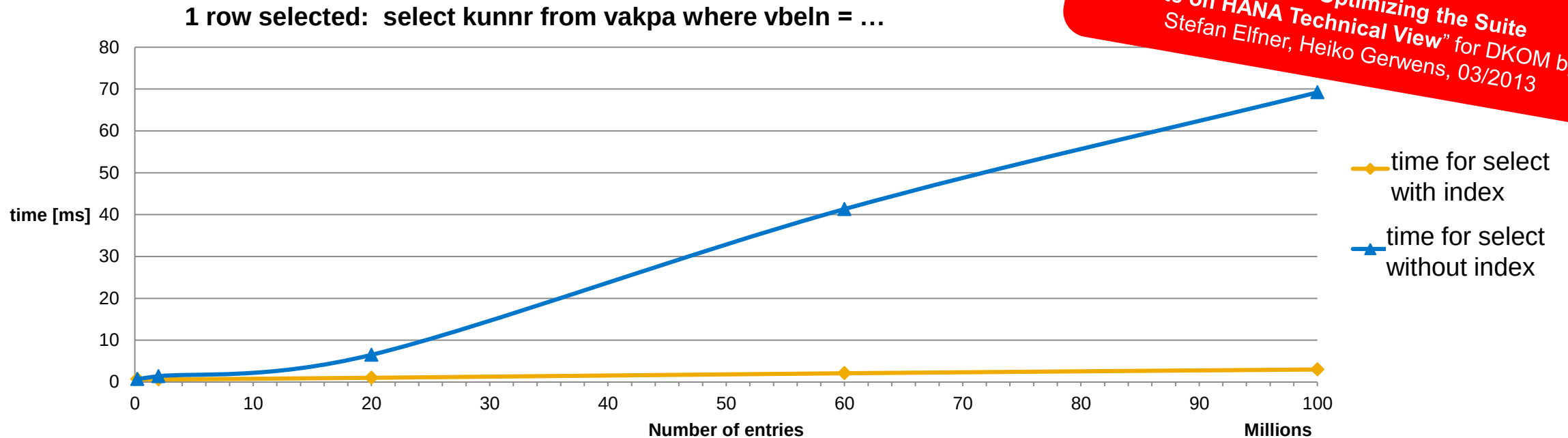
Differences to classic indexes

- **Inv. indexes are dense (just like the main vector).**
That means: no possibility for wasting space
- **Inv. indexes are read only (just like the whole main-part):**
That means: no index reorganizations required, ever.
- Usage of inv. Indexes **can** be seen in PlanViz as of SPS 7

Usage scenarios - Suite on HANA

- **Suite on HANA**
 - Many selective lookups on large tables
 - SAP Note [1794297 - Secondary Indexes for the business suite on HANA](#) delivers ABAP report to create secondary indexes based on heuristic:
 - Create a **single column secondary index** (inverted index) on column store tables for
 - **Small** tables (up to 500.000 records),
column to be indexed must have at least 10.000 distinct values
 - **Large** tables (larger than 500.000 records),
column to be indexed must have at least 1.000 distinct values
 - Also: **(#of records/#of distinct values)** needs to be smaller or equal 10.000, as the filtered result set shouldn't be larger than 10.000 records
 - In other words: a potential index should be used **mainly for larger tables** and **should be able to reduce the result set** for queries with a precise filter **by a large factor**
 - There will be no indexes on time fields (ABAP data type!), “... *because they are almost always selected with ranges with poor selectivity...*”

Sidestep Indexes on HANA: Indexes for HANA will be required



Stolen Slide!
Original see "Optimizing the Suite Suite on HANA Technical View" for DKOM by Stefan Elfner, Heiko Gerwens, 03/2013

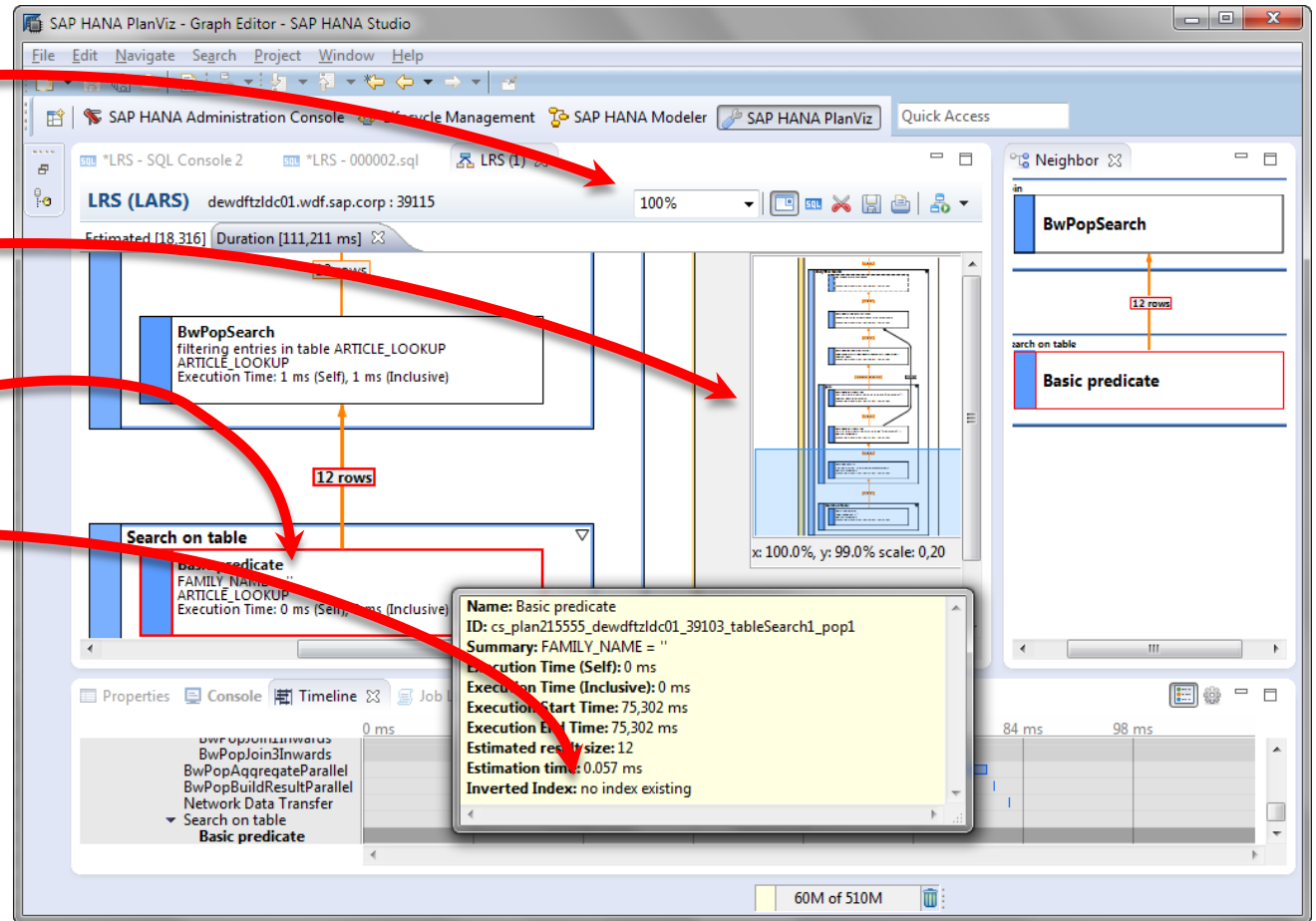
- For very large tables without index: Fast, but still too slow for OLTP type accesses
- Very frequent queries via very selective attributes on large tables still need secondary indexes!

{DEMO}

PlanViz

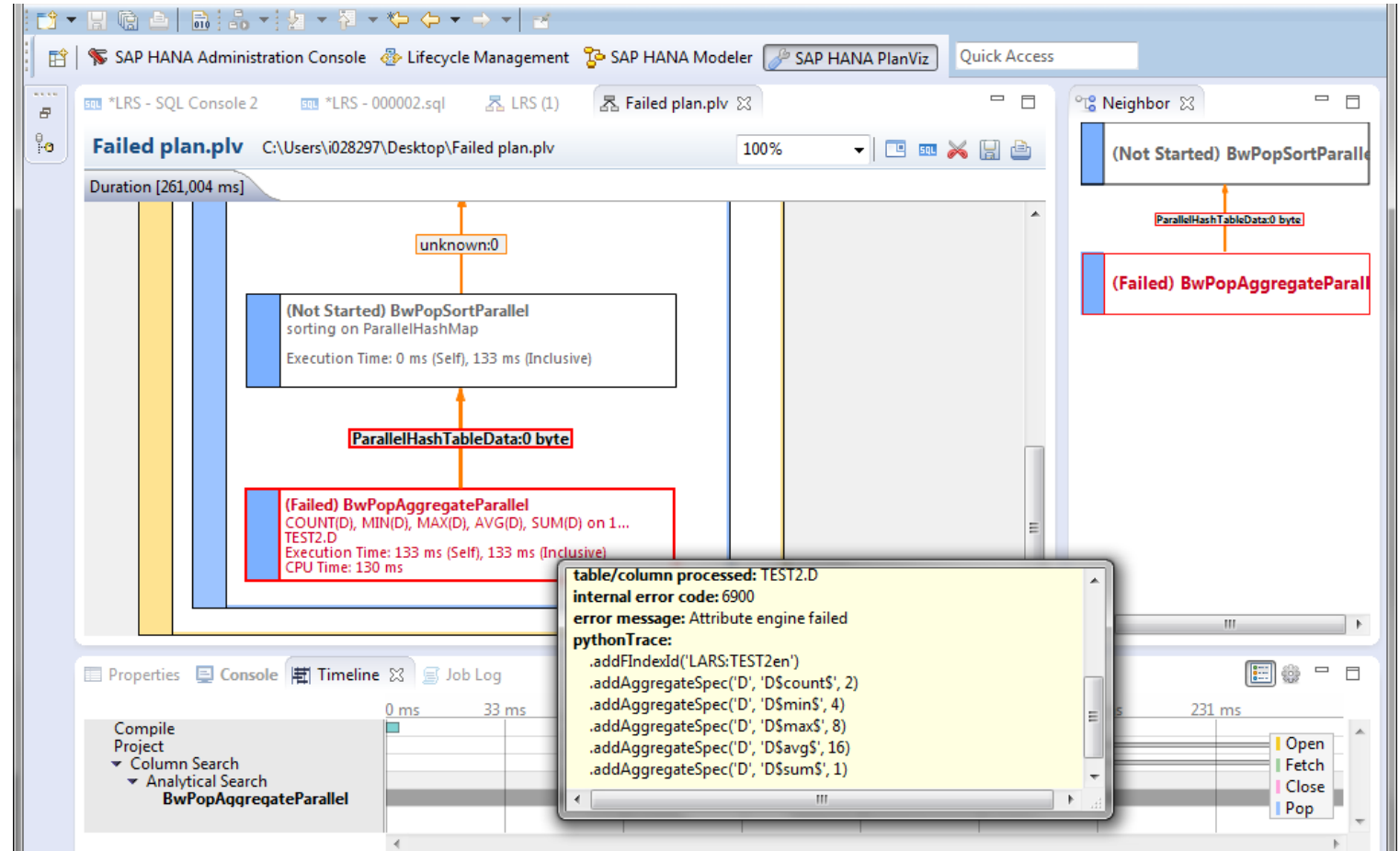
PlanViz UI enhancements

- Predefined PlanViz perspective
- Zoom-Level box
- “Neighborhood” view
- Applied filters visible in POP description
- Inverted Index information in tooltip



PlanViz enhancements

- Failed plans can still be reviewed
- Engine error code in tooltip



{DEMO}



Thank you

Lars Breddemann

RIG Expert // Strategic Customer Engagements (SCE) // Products & Innovation (P&I)
lars.breddemann@sap.com

Submit SCE Feedback: SCE_feedback@sap.com