Usage of in-memory column-based SAP HANA databases in enterprise information systems

PUBLIC

Radim Benek, SAP
October, 2018

THE BEST RUN SAP

SAP Labs Czech Republic in Brno

SAP ČR in Brno from 1995, SAP Labs from 2016

S/4 HANA and Cloud development

delivers innovative cross-platform web applications that are based on modern design principles and technologies.

Globalization Services

focus on developing country specific functions for SAP financial solutions

Application Innovation Services

supports all companies running SAP solutions with a strong focus on continuous innovation

Agenda

In-memory column store database

Examples

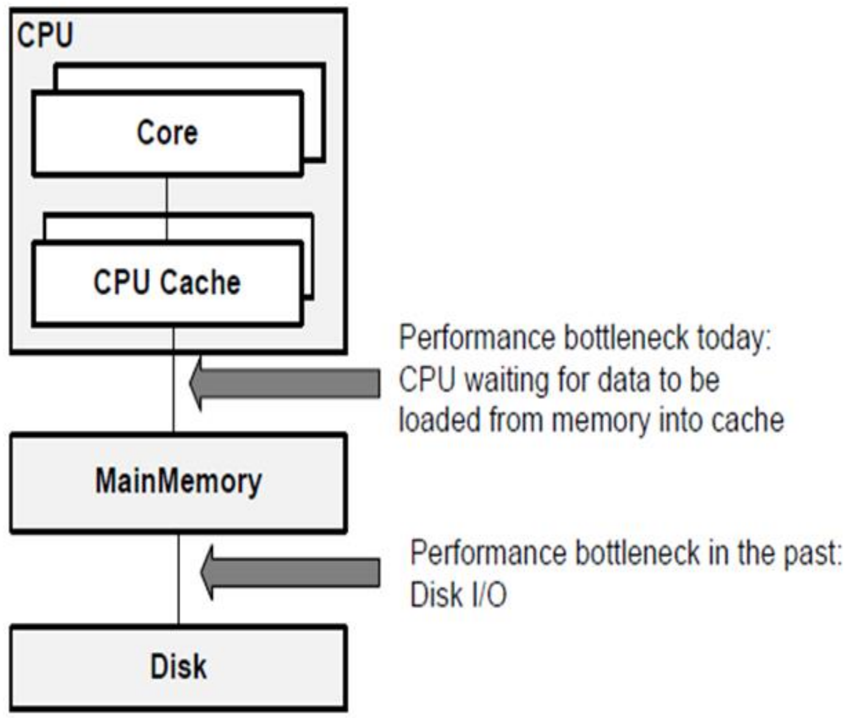
Additional features

SAP HANA Platform

In-memory column store
database

Changes in Hardware

Performance bottleneck



Type of Memory	Size	Latency (~)
L1 CPU Cache	64K	1 ns
L2 CPU Cache	256K	5 ns
L3 CPU Cache	8M	20 ns
Main Memory	GBs up to TBs	100ns
Disk	TBs	>1.000.000 ns

What is SAP HANA?

An Appliance of Hard- and Software

Hardware-Innovations

Multi-core Architecture
(8 x 10 core CPU / blade)

Massive parallel scaling
with many blades

64-bit address space – 2TB
in current servers

100 GB/s data throughput

Dramatic decline in
price/performance

Software-Innovations

Row and column store

Compression

Partitioning

No aggregate
tables

Insert only on delta

Dictionary Encoding
Example

- 8 billion humans
- Each attribute is dictionary encoded

recID	fname	lname	gender	city	country	birthday
…	…	…	…	…	…	…
39	John	Smith	m	Chicago	USA	12.03.1964
40	Mary	Brown	f	London	UK	12.05.1964
41	Jane	Doe	f	Palo Alto	USA	23.04.1976
42	John	Doe	m	Palo Alto	USA	17.06.1952
43	Peter	Schmidt	m	Potsdam	GER	11.11.1975
…	…	…	…		…	…

Dictionary Encoding

Dictionary Encoding a Column

- A column is split into a dictionary and an attribute vector
- Dictionary stores all distinct values with implicit valueID
- Attribute vector stores valueIDs for all entries in the column
- Position is stored implicitly
- Enables offsetting with bit-encoded fixed-length data types

recID	fname
…	…
39	John
40	Mary
41	Jane
42	John
43	Peter
…	…

Dictionary for "fname"

valueID	Value
…	…
23	John
24	Mary
25	Jane
26	Peter
…	…

Attribute Vector for "fname"

position	valueID
…	…
39	23
40	24
41	25
42	23
43	26
…	…

Dictionary Encoding
Data Size Examples

Column	Cardi-nality	Bits Needed	Item Size	Plain Size	Size with Dictionary (Dictionary + Column)	Compression Factor
First names	5 millions	23 bit	50 Byte	400GB	250MB + 23GB	≈ 17
Last names	8 millions	23 bit	50 Byte	400GB	400MB + 23GB	≈ 17
Gender	2	1 bit	1 Byte	8GB	2b + 1GB	≈ 8
City	1 million	20 bit	50 Byte	400GB	50MB + 20GB	≈ 20
Country	200	8 bit	47 Byte	376GB	9.4kB + 8GB	≈47
Birthday	40000	16 bit	2 Byte	16GB	80kB + 16GB	≈ 1
Totals			**200 Byte**	**≈ 1.6TB**	**≈ 92GB**	**≈ 17**

Compression

Compression Techniques

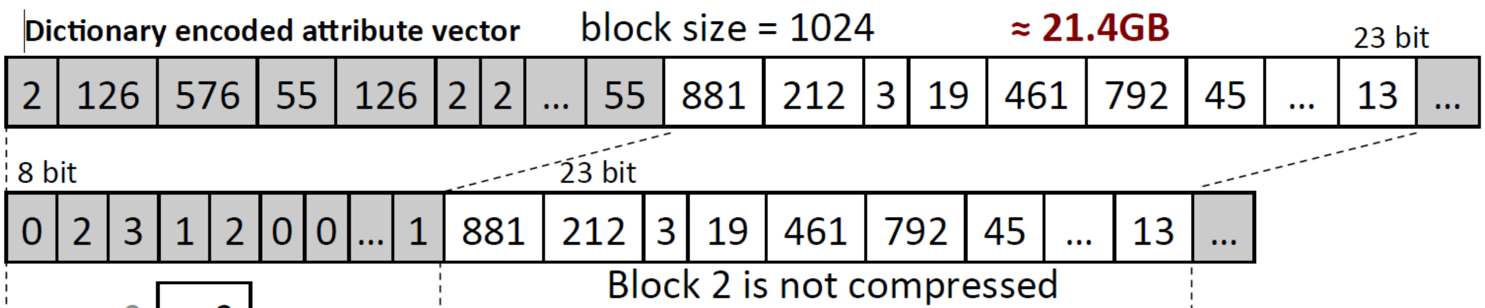
- For attribute vector
  - –Prefix encoding
  - –Run length encoding
  - –Cluster encoding
  - –Sparse encoding
  - –Indirect encoding
    - Sequence is partitioned into *N* blocks of size *S* (typically 1024)
    - If a block contains only a few distinct values an additional dictionary is used to encode the values in that block
    - Additionally: links to the new dictionaries + blocks that have a dictionary

Compression

Indirect Encoding

Example: fname column, table sorted by country

Direct access!



0	2
1	55
2	126
3	576
…	…
199	

23 bit

Dictionary for Block 1

Assumption: each set of 1024 people of the same country contains on average 200 different names

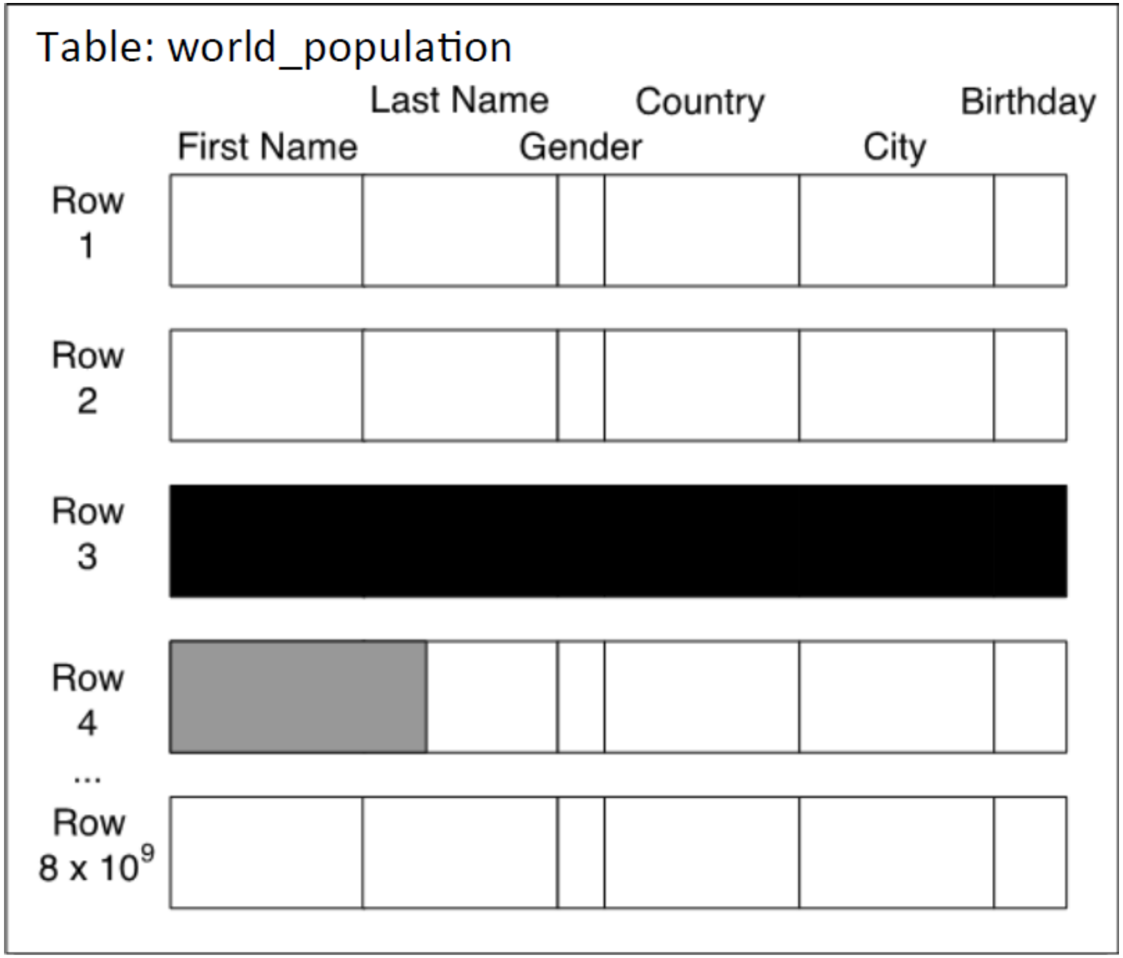
Dictionaries: (200 × 23 bit+64 bit) × #blocks 4.2 GB

Address of dictionary (7.8 mio)

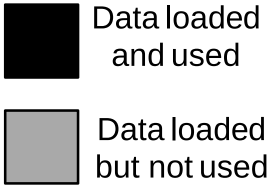
Compressed vector: 8 billion × 8 bit 7.6 GB ≈ **11.8 GB**

Tuple Reconstruction

Row store

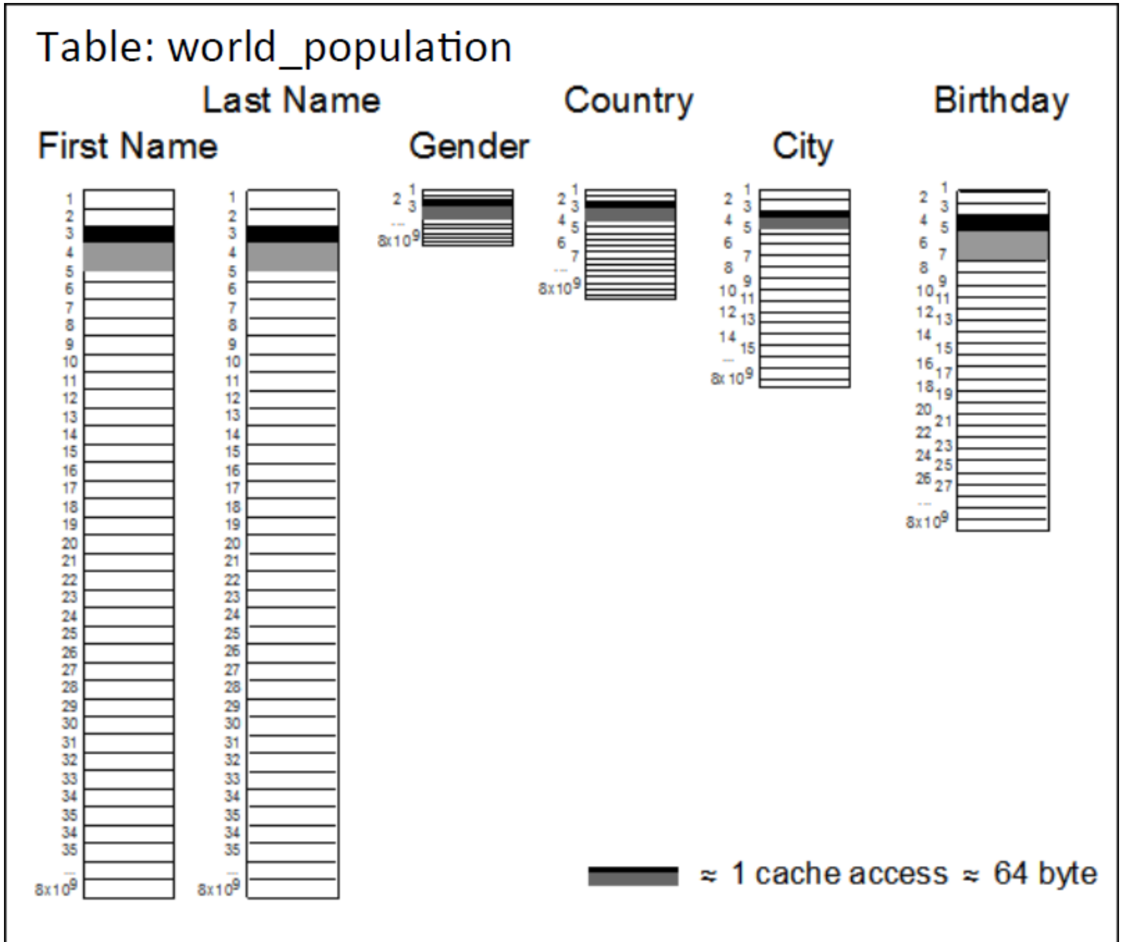


- All attributes are stored consecutively
- 200 byte → 4 cache accesses à 64 byte → 256 byte
- Read with 4MB/ms/core
- → ≈ **0.064 μs** with 1 core



Tuple Reconstruction

Column Store



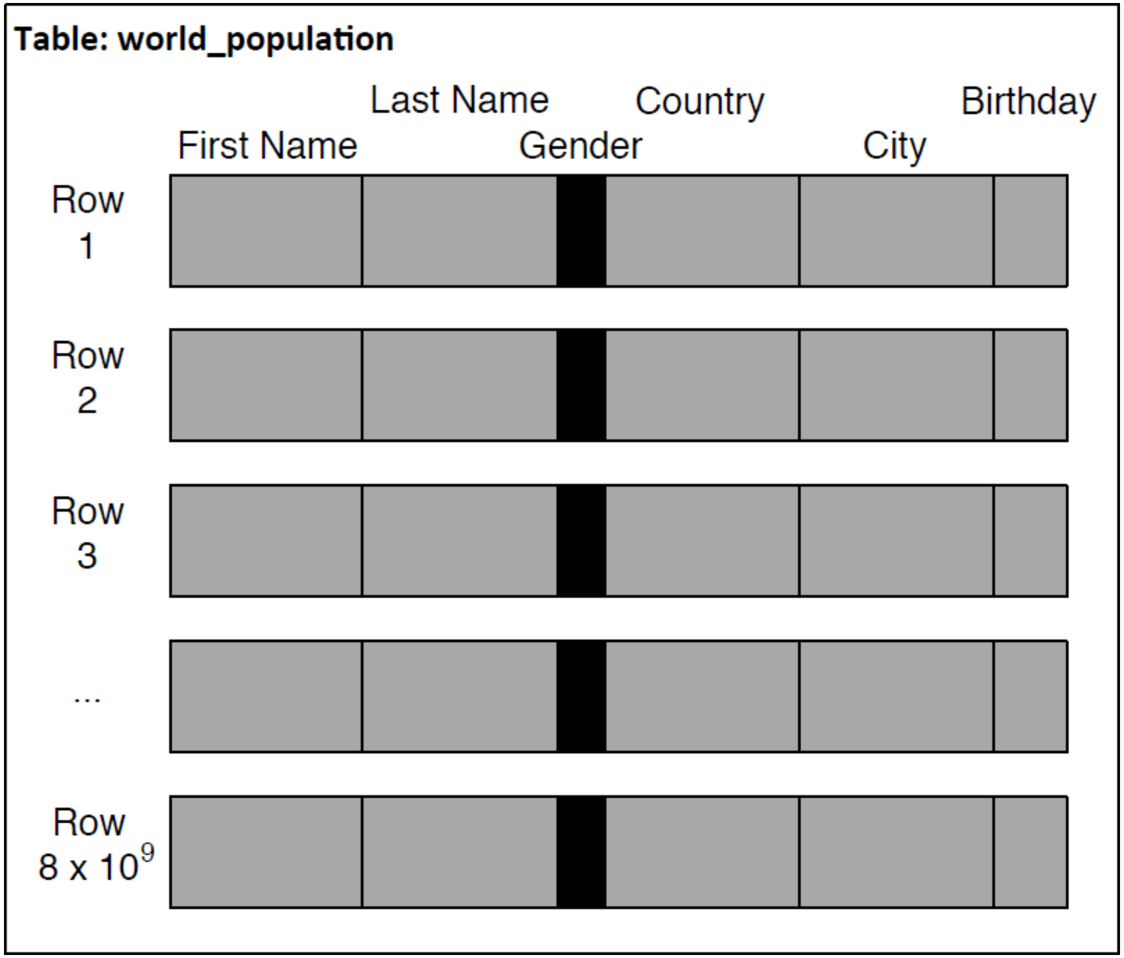
- 1 cache access for each attribute
- 6 cache accesses à 64 byte
  → 384 byte
- Read with 4MB/ms/core
- → ≈ **0.096 µs** with 1 core

Data loaded and used

Data loaded but not used

Scan Performance

Row Store – Full Table Scan



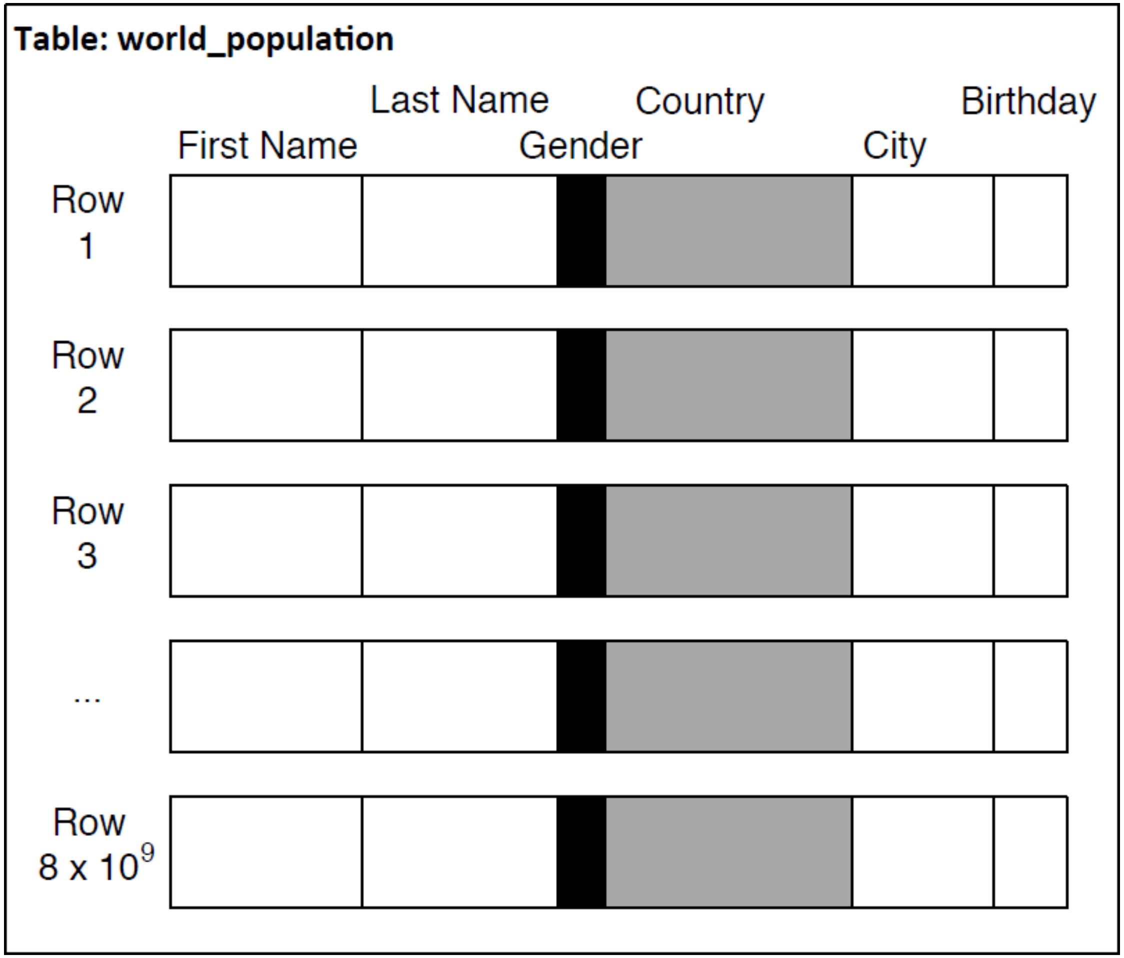
- Table size 8 billion tuples × 200 bytes per tuple
  ≈ 1.6 TB
- Scan through all rows with 4 MB/ms/core
  → **400 s** with 1 core

Data loaded and used

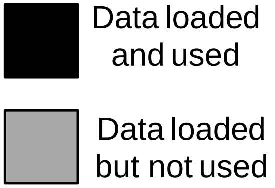
Data loaded but not used

Scan Performance

Row Store – Stride Access „Gender“

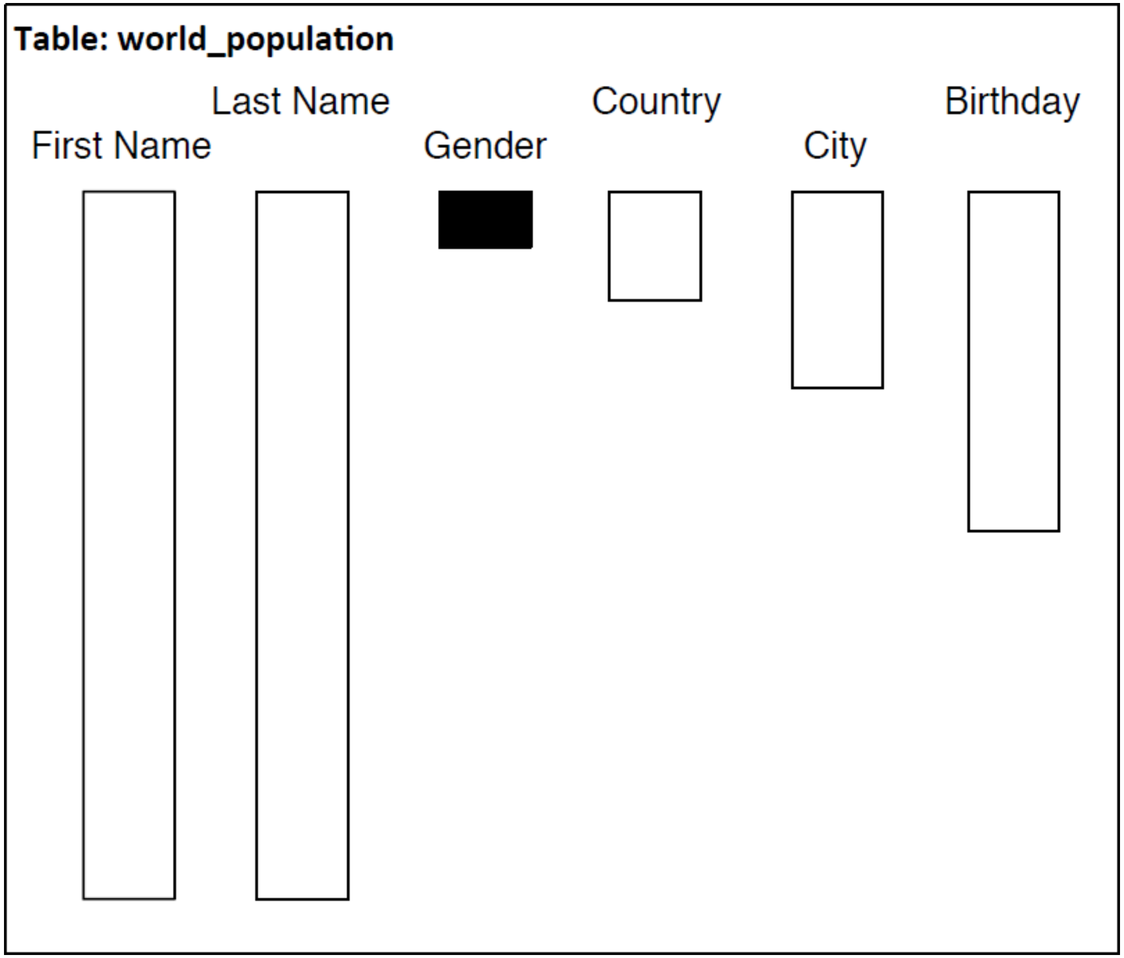


- 8 billion cache accesses à 64 byte ≈ 512 GB
- Read with 4 MB/ms/core → 128 s with 1 core

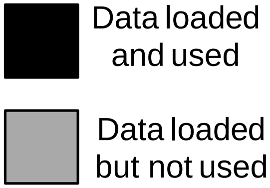


Scan Performance

Column Store – Full Column Scan „Gender“



- Size of attribute vector "Gender":
  8 billion tuples × 1 bit per tuple
  ≈ 1 GB
- Scan through column with 4MB/ms/core
  → 0.25 s with 1 core



Database Operations

INSERT – example (With New Dictionary Entry)

INSERT INTO world_population **VALUES** (Karen, Schulze, f, GER, Rostock, 06-20-2014)

	AV (old)		AV (new)
0	2	0	3
1	3	1	4
2	1	2	1
3	0	3	0
4	4	4	5

	D (new)
0	Anton
1	Hanna
2	Karen
3	Martin
4	Michael
5	Sophie

	fname	lname	gender	country	city	birthday
0	Martin	Albrecht	m	GER	Berlin	08-05-1955
1	Michael	Berg	m	GER	Berlin	03-05-1970
2	Hanna	Schulze	f	GER	Hamburg	04-04-1968
3	Anton	Meyer	m	AUT	Innsbruck	10-20-1992
4	Sophie	Schulze	f	GER	Potsdam	09-03-1977
5		Schulze			Rostock	
...	...	...	...	...	...	...

1. Look-up on dictionary → no entry found
2. Append new value to dictionary
3. Sort Dictionary
4. Change valueIDs in attribute vector
5. Append new valueID to attribute vector

AV – Attribute Vector
D – Dictionary

Database Operations

DELETE - example

DELETE FROM world_population **WHERE** fname = “Jane” and lname = “Doe”

Dictionary “fname”

valueID	value
...	...
22	Andrew
23	Jane
24	John
25	Mary
26	Peter
...	...

Attribute Vector “fname”

recID	valueID
...	...
38	22
39	24
40	25
41	23
42	24
43	26
...	...

Dictionary “lname”

valueID	value
...	...
17	Brown
18	Doe
19	Miller
20	Schmidt
21	Smith
...	...

Attribute Vector “lname”

recID	valueID
...	...
38	19
39	21
40	17
41	18
42	18
43	20
...	...

Database Operations
UPDATE

UPDATE world_population **SET** city = „Bamberg“
WHERE fname = “Hanna” **AND** lname = “Schulze”

recID	fname	lname	gender	country	city	birthday
0	Martin	Albrecht	m	GER	Berlin	08-05-1955
1	Michael	Berg	m	GER	Potsdam	03-05-1970
2	Hanna	Schulze	f	GER	Hamburg	04-04-1968
3	Anton	Meyer	m	AUT	Innsbruck	10-20-1992
4	Ulrike	Schulze	f	GER	Potsdam	09-03-1977
5	Sophie	Schulze	f	GER	Rostock	06-20-2012
...	...	...	...	...	...	...
8×10^9	Zacharias	Perdopolus	m	GRE	Athen	03-12-1979

Combination of DELETE and INSERT operation

1. Look-up „Bamberg“ in dictionary → entry not found
2. Append new value „Bamberg“ to dictionary
3. Reorganize dictionary
4. Replace old values with new values in attribute vector (expensive)

Examples

Performance measurement
Examples

- System QM0 – 48 TB / 1100 CPUs

Table	Store	Rows	Size	Time
ACDOCA_C	Column	110 million	5 GB	1,8 s
ACDOCA_R	Row	110 million	240 GB	22,5 s
ACDOCA_sm	Column	5 million	140 MB	0,3 s
ACDOCA	Column	**19,5 billion**	1,3 TB	139 s
CDHR	Column	31 million	1,3 GB	12,4 s
CDPOS	Column	730 million	44 GB	

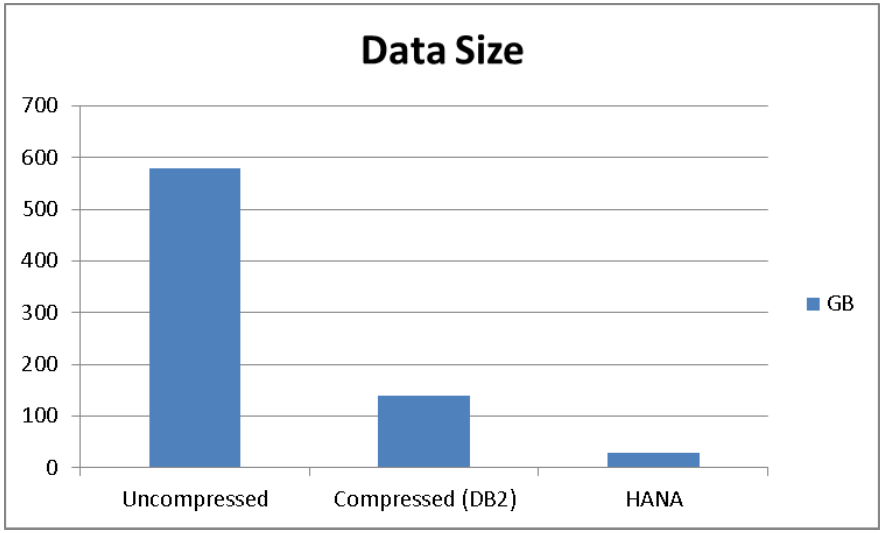
- System HANA Express edition (VM) – 16 GB / 4 CPUs

Table	Store	Rows	Size	Time
ACDOCA_sm	Column	5 million	140 MB	0,9 s

Application improvements

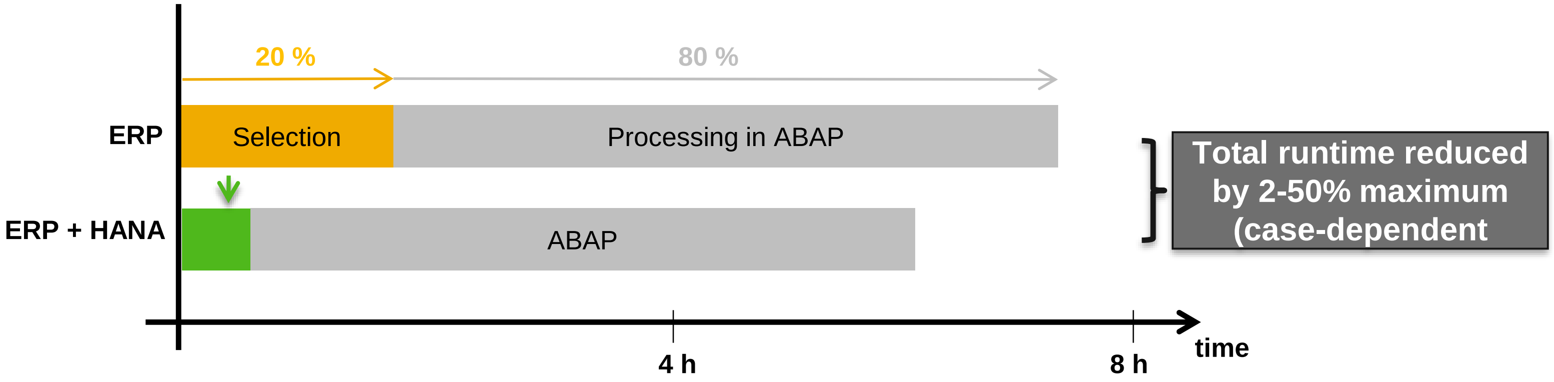
CO-PA Accelerator: Result provided by a Customer

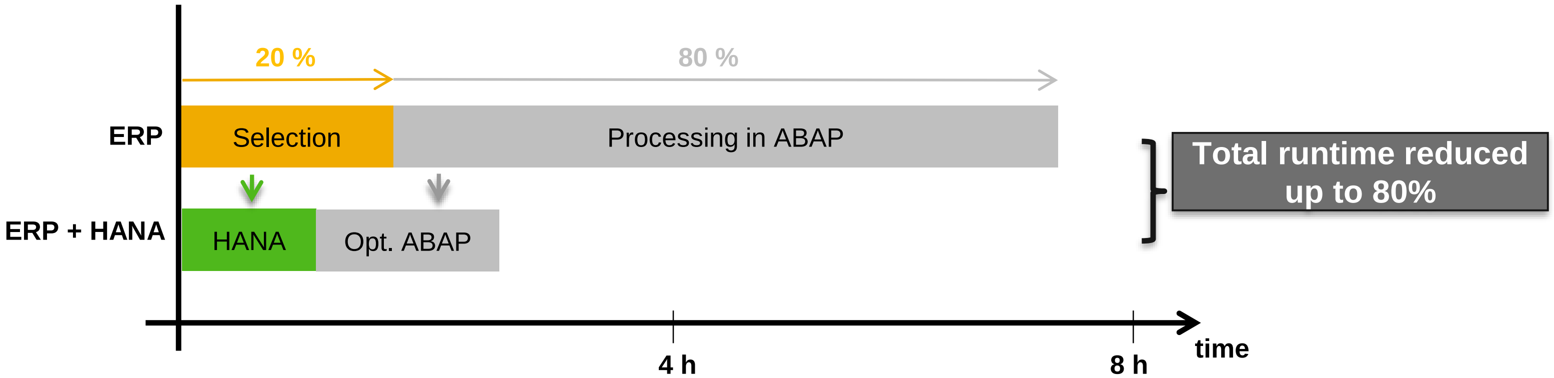
- COPA Accelerator implemented within 8 weeks (including test & production landscape).
- No manual modeling or creation of analytical views in HANA needed.
- Only replication of transactional CO-PA data needed.
- Some figures about data volume:
  - Total records in HANA: 550 Mil
  - Total volume in HANA: 30 GB
  - Total volume in ERP DB2:
    - 580 GB uncompressed,
    - 140 GB compressed (on disk)
- Number of posted records/day:
  - 100.000 - 200.000
- Initial replication took ~24 hours



Application improvements

CO-PA Accelerator: Top Down Accelerating Period-End Closing





Application improvements

CO-PA Accelerator: KE28 – Validation with Productive Customer-Data

CO-PA Data ▪ ~ 350 Mil Line Items ▪ ~ 80 Mil CE4-Items	**KE28 w/o SAP HANA**	**HANA-optimized KE28**	**Acceleration in Factors**	**Acceleration in %**
Top-Down-Distribution Level 1 • 6 Variants with Postings	5.880 sec	184 sec	32	**97 %**
• 10 Variants without Postings	7.550 sec	194 sec	39	**97 %**
Top-Down-Distribution Level 2 • 13 Variants with Postings	25.096 sec	13.282 sec	2	**50 %**
• 181 Varianten without Postings	64.557 sec	1.782 sec	36	**97 %**
Total Runtime	**28,6 h**	**4,3 h**	**7**	**86 %**

- Existing KE28 Variants will be accelerated with no changes to customizing or job-scheduling up to factor 40
- Significant unload of primary DB during period-end closing activities

Additional features

Backup/Recovery

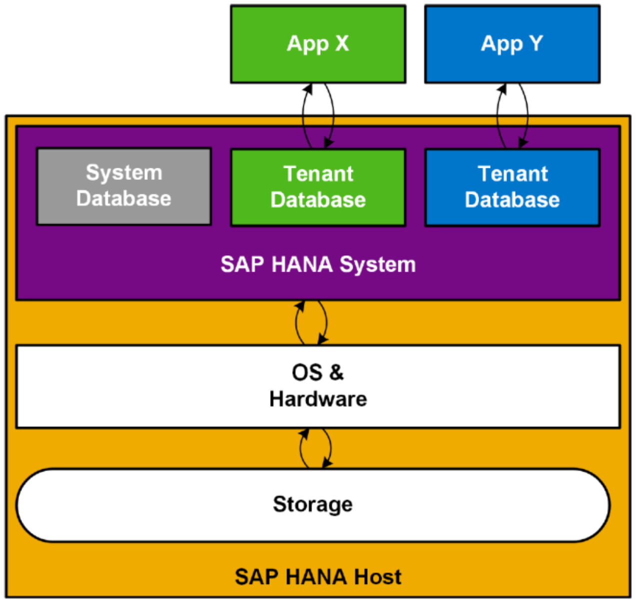
SAP HANA holds the bulk of its data in memory for maximum performance, but still uses persistent storage to provide a fallback in case of failure.

- During normal operation, data is automatically saved from memory to disk at regular savepoints. Additionally, all data changes are captured in redo log entries. A redo log entry is written to disk after each committed database transaction.
- Support for multitenant database containers
- Apply to all of SAP HANA, both the hot and the warm store
- Backint enables 3rd party tool vendors to directly connect backup agents

Multitenant and scale

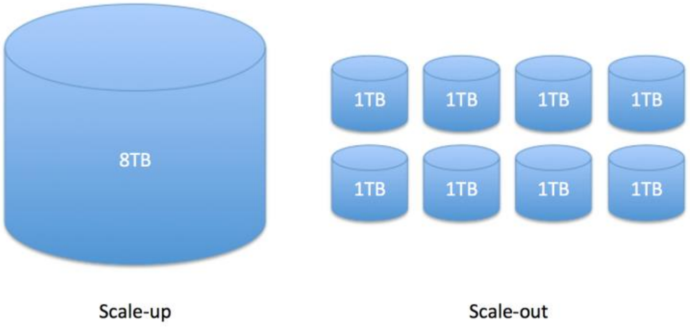
Multitenant Data Base

- A single SAP HANA can contain several isolated databases
- The tenant databases share computing resources (RAM, CPU), SW installation, system administration (start/stop system).
- The tenant database has their own metadata, data, and users.



Scale the system

- More data -> more RAM -> more CPUs
- Scale up: one system up 20 CPU and 20 TB RAM.
- Scale out: combining multiple independent nodes into one system (supporting multitenant).

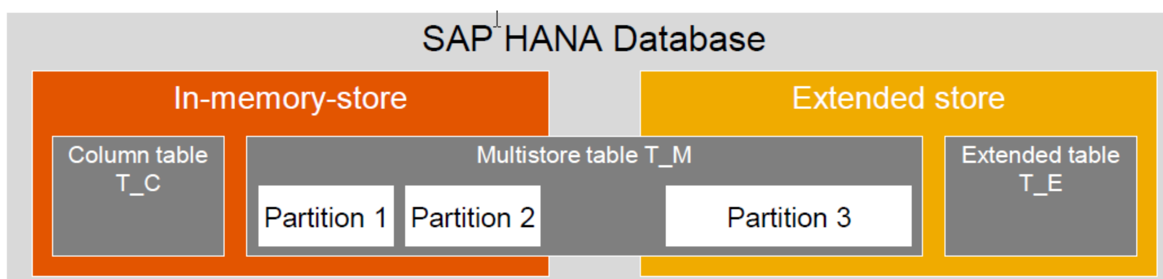


Dynamic Tiering

Data temperatures

Data tiering is the assignment of data to various tiers/storage media based upon data type, performance requirements, frequency of access.

SAP HANA In-Memory Store	Hot Store	This tier is used to store mission-critical data for real-time processing and real-time analytics. Data is retained "in-memory".
Data Tiering with Integrated Storage	Warm Store	This tier is used to store data with reduced performance SLAs, which is less frequently accessed. Data is stored on a lower cost storage tier, managed as a unified part of the SAP HANA database.
Data Tiering with External Storage	Cold Store	This tier is used to store voluminous data for sporadic or very limited access. Data is stored on disk, in columnar structures on SAP IQ or in Hadoop HDFS.



SAP HANA Platform

SAP HANA: The business data platform for the intelligent enterprise

SAP HANA PLATFORM

APPLICATION DEVELOPMENT

- Web Server
- JavaScript
- Fiori UX
- Graphic Modeler
- Application Lifecycle Management

ADVANCED ANALYTICAL PROCESSING

- Spatial
- Graph
- Predictive
- Search
- Text Analytics
- Streaming Analytics
- Series Data
- Business Functions

DATA INTEGRATION & QUALITY

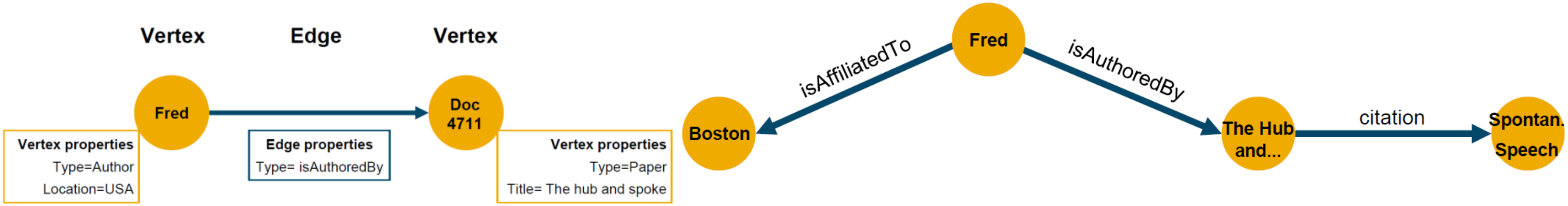
- Data Virtualization
- ELT & Replication
- Data Quality
- Hadoop & Spark Integration
- Remote Data Sync

DATABASE MANAGEMENT

- Columnar OLTP+OLAP
- Multi-Core & Parallelization
- Advanced Compression
- Multi-tenancy
- Multi-Tier Storage
- Data Modeling
- Openness
- Admin & Security
- High Availability & Disaster Recovery

Graph
Definition

- The property graph model provides directed, attributed multi-relation graphs
- Use cases
  – Social network, company/organizations, production and supply chains, citation networks, authorization and role concepts, knowledge graphs...
- SAP HANA supports data graph processing directly in the server.
- Built-in functions like shortest path, get neighborhood, topological analysis of complete graph
- Support for pattern matching using openCypher
- GraphScript to develop custom graph algorithms



Graph

Code example

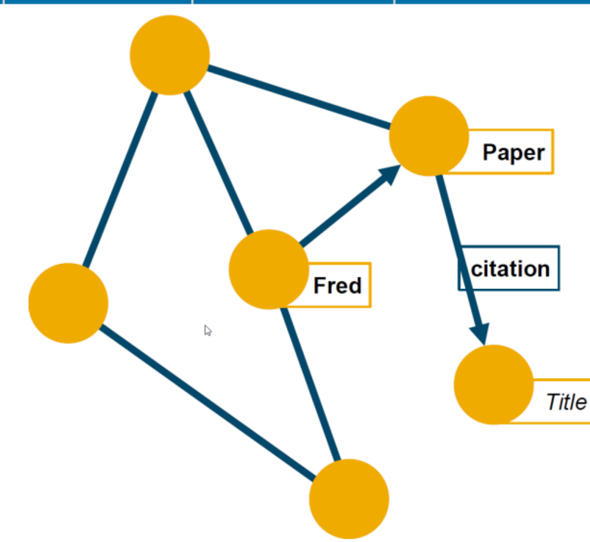
A GRAPH WORKSPACE exposes the data to the graph engine

```
CREATE GRAPH WORKSPACE [SCHEMA].[NAME]
  EDGE TABLE [SCHEMA].[EDGE TABLE/VIEW]
    SOURCE COLUMN source
    TARGET COLUMN target
    KEY COLUMN id
  VERTEX TABLE [SCHEMA].[NODE TABLE/VIEW]
    KEY COLUMN id;
```

ID	TYPE	NAME	YEAR
AUT-6841	Author	Richardson, Fred	
H94-1009	Paper	The Hub and Spoke Paradigm for CSR Evaluation	1994
ORG-523	Organization	Boston University	
H92-1076	Paper	Spontaneous Speech Collection for the CSR Corpus	1992

ID	SOURCE	TARGET	TYPE
1	H94-1009	AUT-6841	isAuthoredBy
2	AUT-6841	ORG-523	isAffiliatedTo
3	H94-1009	H92-1076	citation

- **MATCH** (A) -[e1]->(P1), (P1) -[e2]->(P2)
  WHERE A.NAME = 'Fred'
  AND P1.TYPE = 'Paper'
  AND e2.TYPE = 'citation'
  RETURN P2.TITLE AS TITLE



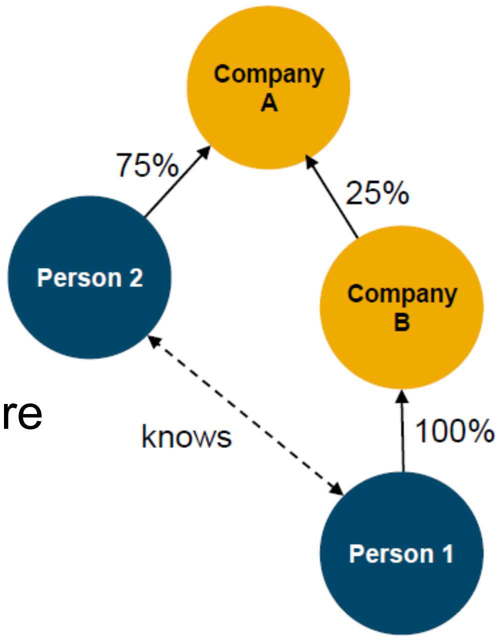
Graph
Customer example

Customer collects and analyze data about companies, people and their connections.

Graph size
- 24 Mio Nodes (organizations, persons)
- 125 Mio. Edges (owns, knows, etc.)

UBO (ultimate beneficial owner) description
- All persons owning 25% or more of a company are UBO
- All persons “controlling” a company which owns 25% or more are UBO

UBO implementation with HANA Graph
- ~15 lines of code
- 5 minutes to identify all UBOs for millions of companies

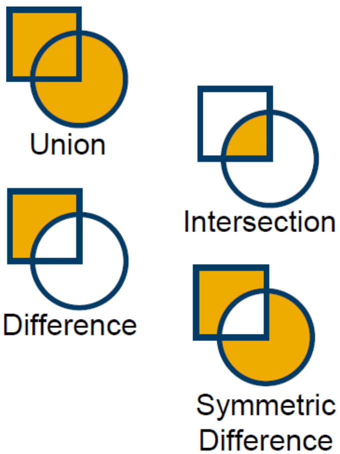


Spatial data

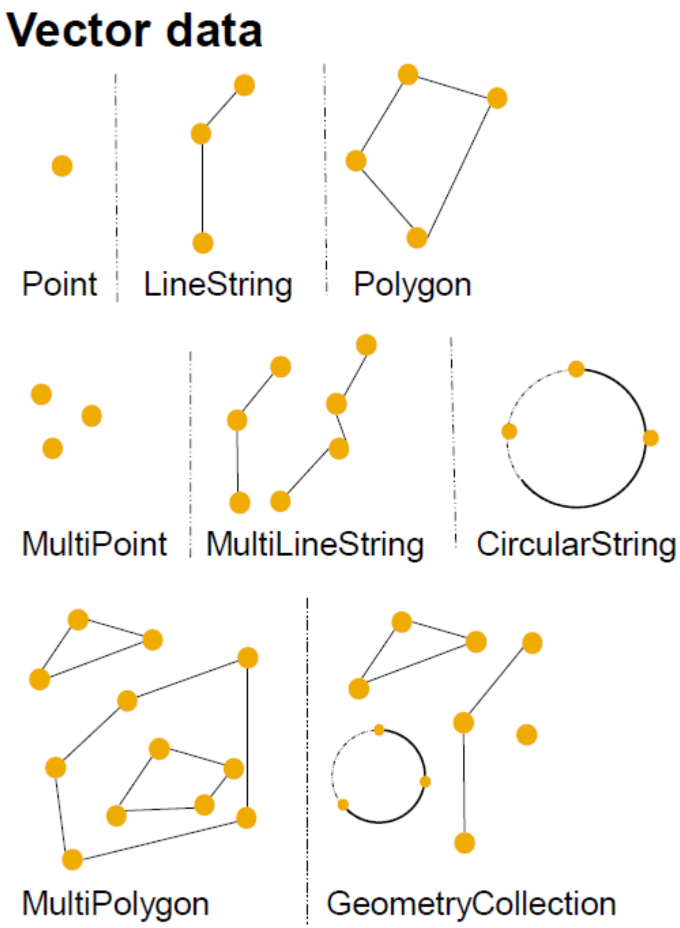
Types and functions

SAP HANA provides native spatial data processing

- Natively store 2D, 3D and 4D vector data types (x, y, z, m)
- Over 80 native SQL based geospatial functions
- Open standards (OGC, 1999 SQL/MM)



```
"POINT".ST_Within("RECTANGLE") = 1
```



Spatial data

Code example

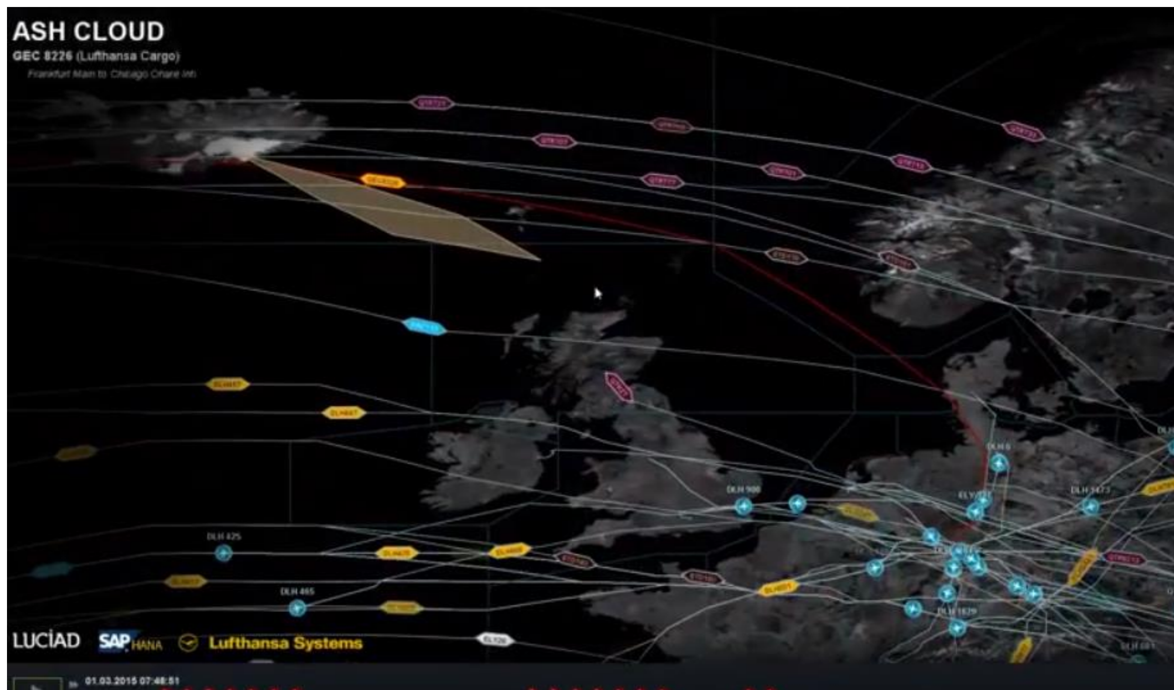
- **CREATE COLUMN TABLE** shapes (
  id **BIGINT,**
  description **NVARCHAR(IOO),**
  shape **ST_GEOMETRY(4326)**);

- **INSERT INTO** shapes **VALUES** (1, ’a', new ST_Point(‘POINT(1.6, 2.0)’, 4326));
- **INSERT INTO** shapes **VALUES** (3, ’c’, new ST_Polygon(’Polygon((0 0, 1 0, 1 1,0 1,0 0))’, 4326));

- **SELECT** id, description, shape.ST_AsSVG(), shape.ST_Area()
  FROM shape
  WHERE shape.ST_Intersects(new ST_POLYGON(
  ‘POLYGON((0 0, 3 0, 3 3, 0 3, 0 0))’)) = 1
  AND shape.ST_GeometryType() = ‘ST_Polygon’;

Spatial data
Customer example

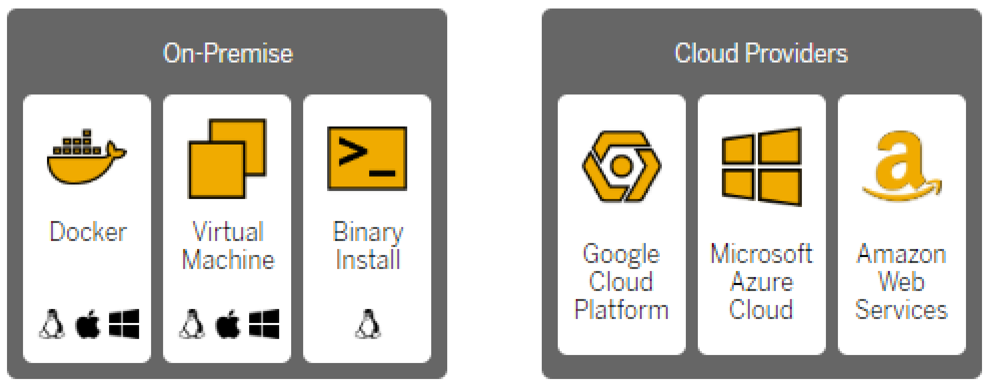
- Airlines need real-time insights into flight operations of several thousand flights per day and be situational-aware of meteorological conditions which can result in cancellations or delay of flights. Airlines need the ability to manage airline traffic in real-time with a global view and provide decision-support to flight dispatchers and pilots to find alternative trajectories while minimizing costs



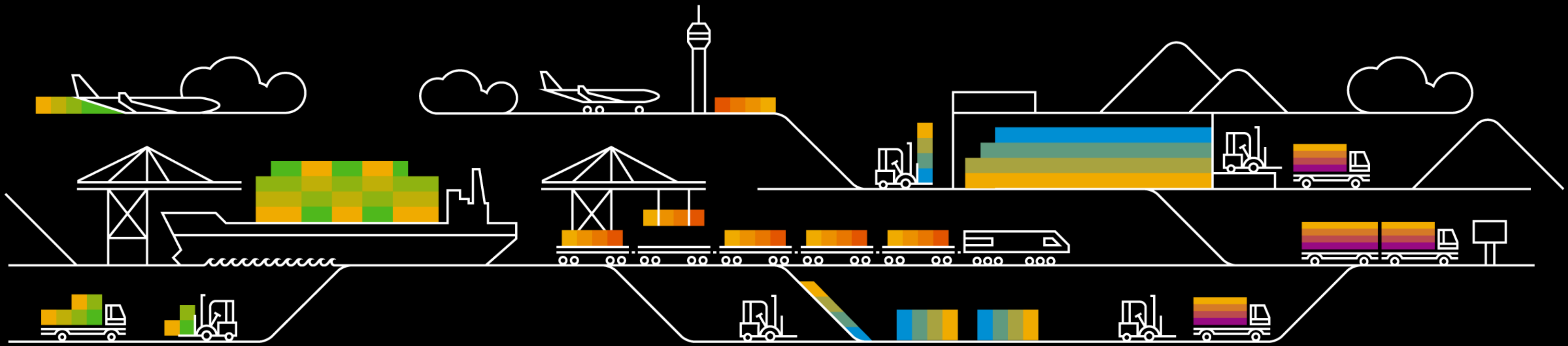
10x
faster

SAP HANA, express edition

SAP HANA, express edition is a database and application development platform. You can run it for free (up to 32GB of RAM) on your laptop and start building new apps.



Resources



Resources

- Plattner, Hasso. “In-Memory Data Management 2015“ OpenHPI. Hasso-Plattner-Institute, 07 Sept. 2015. Web. 13 July 2017. https://open.hpi.de/courses/imdb2015
- Fath, Markus. “Spatial Analysis with SAP HANA Platform“ openSAP, 25 April 2017. Web. 03 October 2018. https://open.sap.com/courses/hsgs1
- Fath, Markus. “Analyzing Connected Data with SAP HANA Graph“ openSAP, 20 June 2018. Web. 03 October 2018. https://open.sap.com/courses/hsgra1
- SAP HANA Academy Videos: https://www.youtube.com/user/saphanaacademy
- SAP Help Portal - SAP HANA Platform: https://help.sap.com/viewer/product/SAP_HANA_PLATFORM/
- SAP HANA, express edition: https://www.sap.com/developer/topics/sap-hana-express.html

Thank you.

Contact information:

Radim Benek
Development Expert, SAP Labs Czech Republic
SAP CR, spol. s r.o.
Holandská 6/8
639 00 Brno
radim.benek@sap.com

THE BEST RUN SAP

Follow us

www.sap.com/contactsap

THE BEST RUN SAP