Star Schema Benchmark für SAP HANA

Jan Hofmeier, Kristina Albrecht, Lion Scherer, Marius Jochheim

1. SAP HANA
2. Star Schema Benchmark
3. Benchmarks
4. Cube Präsentation

Gliederung

SAP HANA

High Performance Analytic Appliance

Hauptspeicher statt Festplatte

Spaltenorientiert (Row store möglich)

OLAP + OLTP

Entwicklungsplattform

Speicherzugriffszeiten

Speicherkomponenten in der Systemarchitektur	Größenordnung der Zugriffszeit
Zugriff auf CPU L1-/L2-/ L3 Cache	0,5 / 7,0 / 15 ns
Zugriff auf Hauptarbeitspeicher	100 ns
Zugriff auf Solid-State-Festplatte (SSD)	150.000 ns
Festplattenzugriff	10.000.000 ns

Bildquelle: In-Memory-Datenbank SAP HANA, Peter Preuss, Springer Gabler 2017

In Memory Vorteile

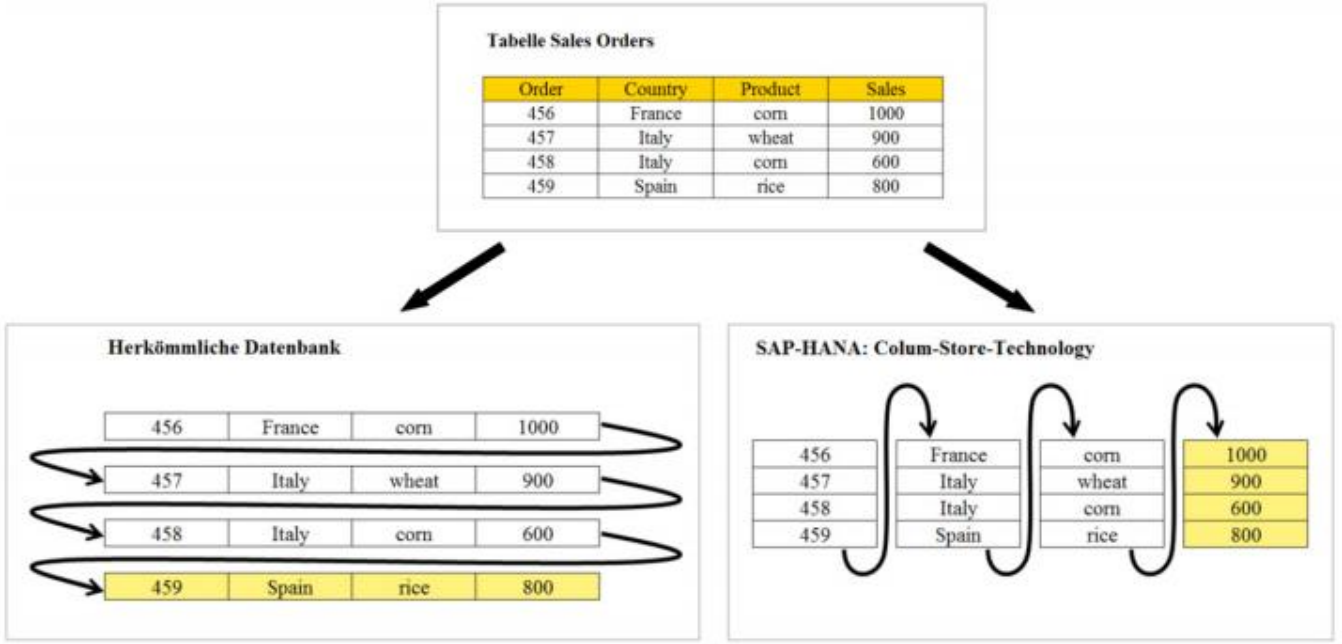
Komplexe Abfragen und Datenbankoperationen mit sehr hohem Durchsatz ausführbar

Kein ETL Prozess

Echtzeit Analysen

Datenzugriffszeit wird verkürzt

Spaltenbasiert vs Zeilenbasiert

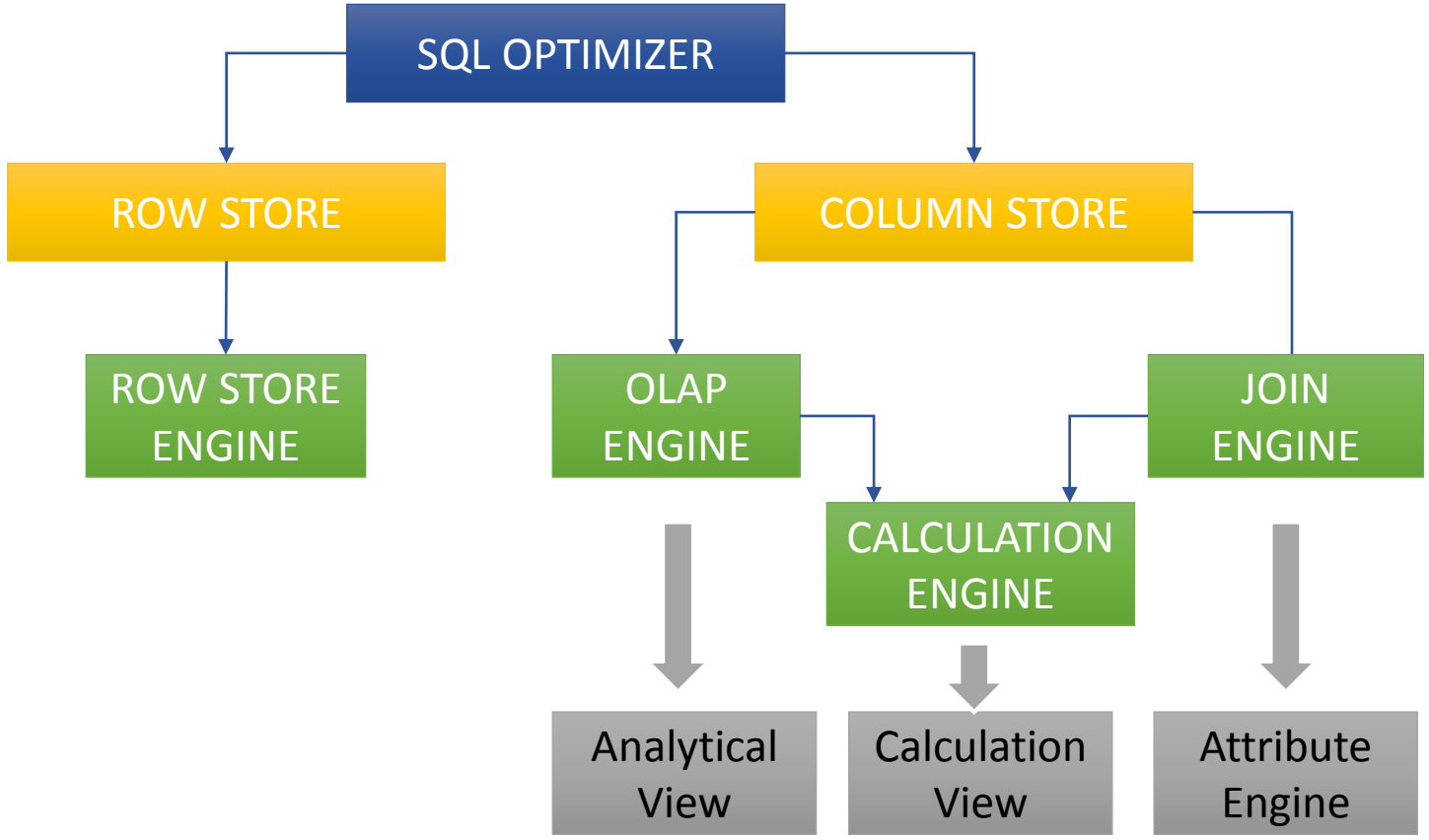


Bildquelle: In-Memory-Datenbank SAP HANA, Peter Preuss, Springer Gabler 2017

Compression

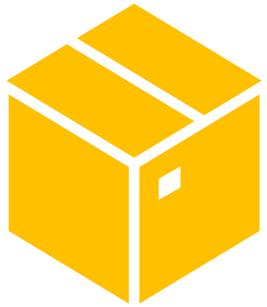
- Dictionary Compression
- Prefix encoding
- Run length encoding
- Cluster encoding
- Sparse encoding
- Indirect encoding

Architektur

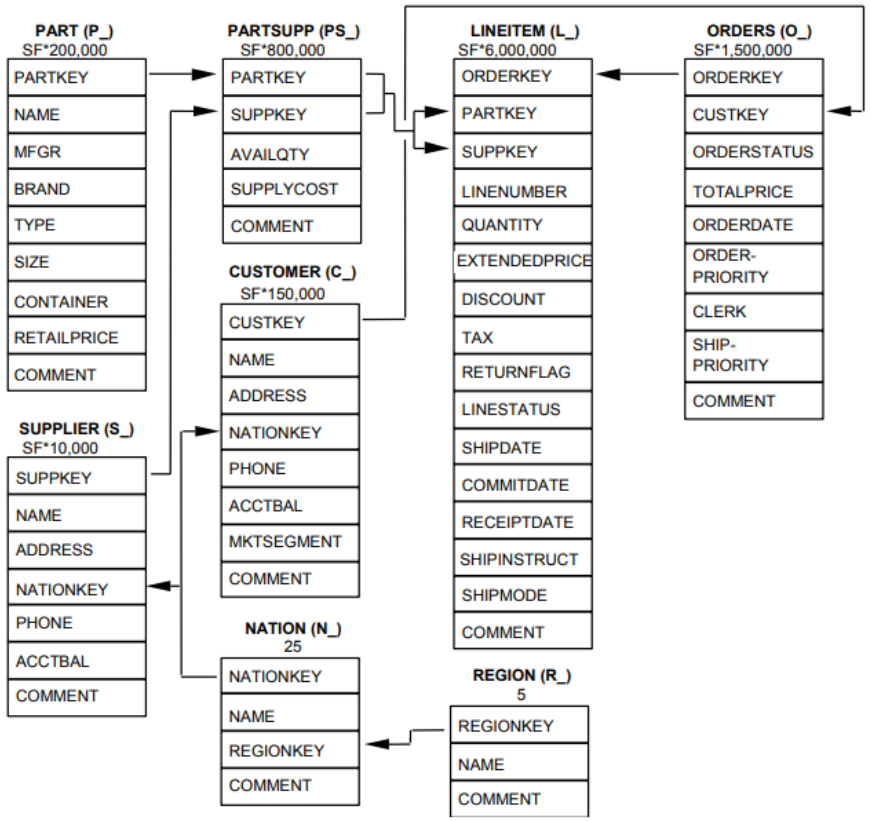


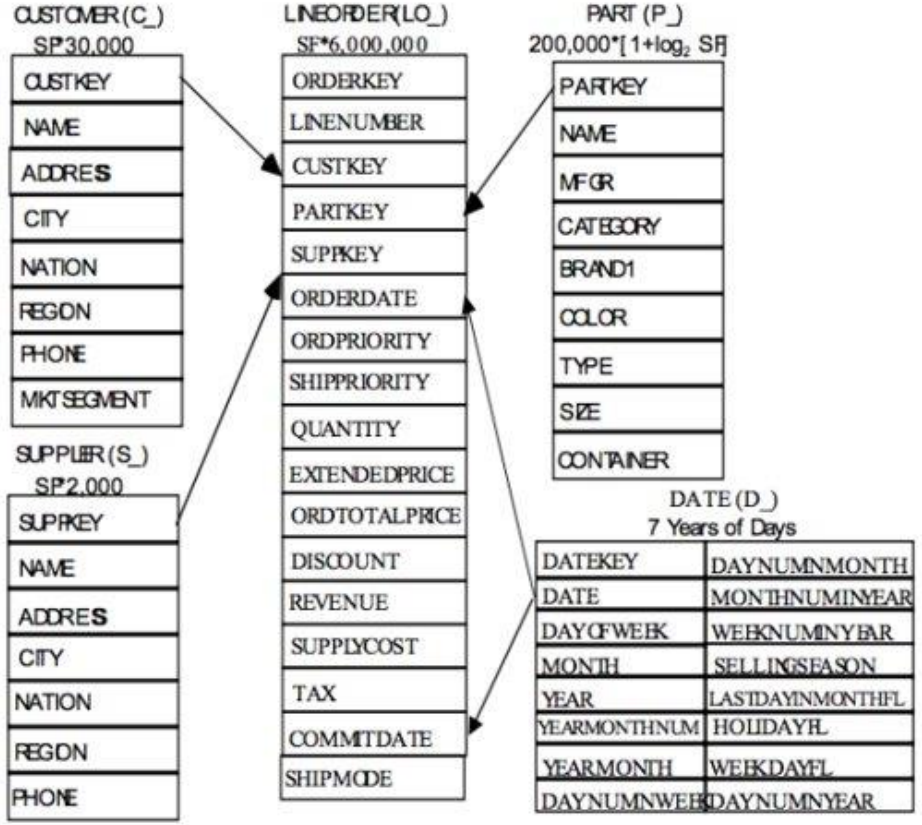
Star Schema Benchmark

9



- Misst die OLAP Performance von Datenbank Produkten
- Basiert auf dem TPC-H Benchmark
- Nutzt ein Star Schema Data Mart mit großer Faktentabelle (LINEORDER) und mehreren Dimenionstabellen





Vergleich von TPC-H und SSB

Group 1

```
SELECT SUM(lo_extendedprice*lo_discount)
AS revenue
FROM lineorder JOIN dim_date
ON lo_orderdatekey = d_datekey
WHERE d_year = 1993
AND lo_discount BETWEEN 1 AND 3
AND lo_quantity < 25;
```

Group 2

```
SELECT SUM(lo_revenue), d_year, p_brand
FROM lineorder
JOIN dim_date ON lo_orderdatekey = d_datekey
JOIN part ON lo_partkey = p_partkey
JOIN supplier ON lo_suppkey = s_suppkey
WHERE p_brand= 'MFGR#2239'
AND s_region = 'EUROPE'
GROUP BY d_year, p_brand
ORDER BY d_year, p_brand;
```

Group 3

```
SELECT c_city, s_city, d_year, SUM(lo_revenue) AS revenue
FROM customer
JOIN lineorder ON lo_custkey = c_customerkey
JOIN supplier ON lo_suppkey = s_suppkey
JOIN dim_date ON lo_orderdatekey = d_datekey
WHERE (c_city='UNITED KI1' OR c_city='UNITED KI5')
AND (s_city='UNITED KI1' OR s_city='UNITED KI5')
AND d_year >= 1992
AND d_year <= 1997
GROUP BY c_city, s_city, d_year
ORDER BY d_year ASC, revenue DESC;
```

Group 4

```
SELECT d_year, s_city, p_brand,
SUM(lo_revenue - lo_supplycost) AS profit
FROM lineorder
JOIN dim_date ON lo_orderdatekey = d_datekey
JOIN customer ON lo_custkey = c_customerkey
JOIN supplier ON lo_suppkey = s_suppkey
JOIN part ON lo_partkey = p_partkey
WHERE s_nation = 'UNITED STATES'
AND (d_year = 1997 OR d_year = 1998)
AND p_category = 'MFGR#14'
GROUP BY d_year, s_city, p_brand
ORDER BY d_year, s_city, p_brand;
```

HANA Performance Benchmark

HANA Performance Benchmark

Test Setup

VM Image von SAP

CPU: Intel(R) Core(TM) i7-6820HQ CPU @ 2.70GHz

CPU Kerne: 4 (x2 Theads)

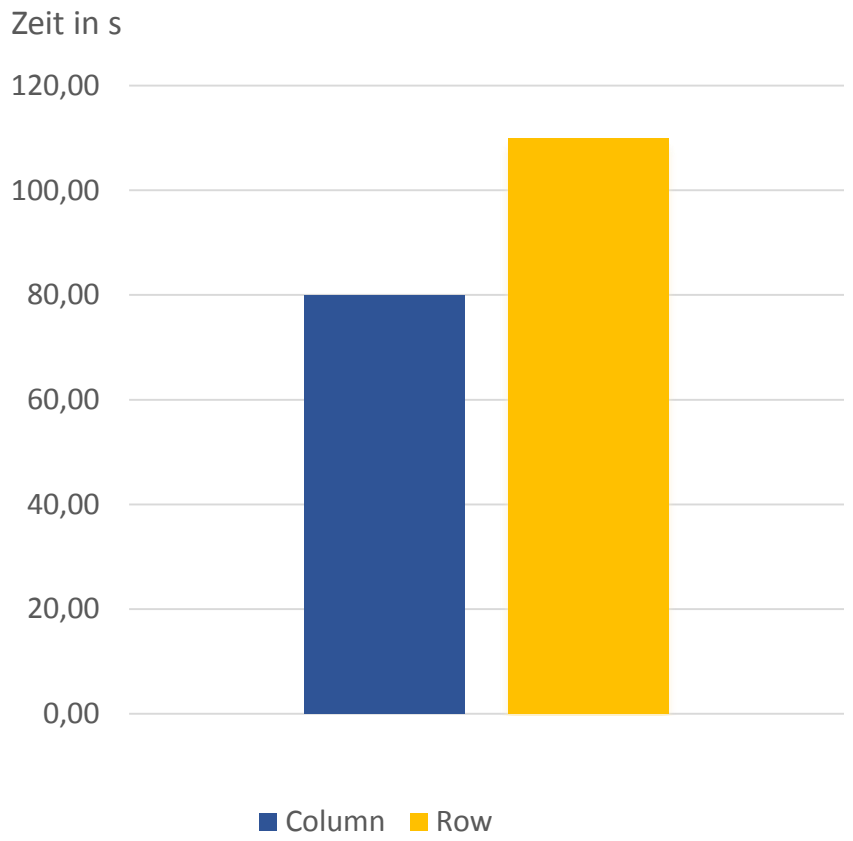
RAM: 16GB

Storage: SSD

HANA Performance Benchmark

Ladezeiten von Tabellen

27 % weniger Zeit



HANA Performance Benchmark

Ladezeiten von Indizes

HANA Performance Benchmark

Vorgehensweise

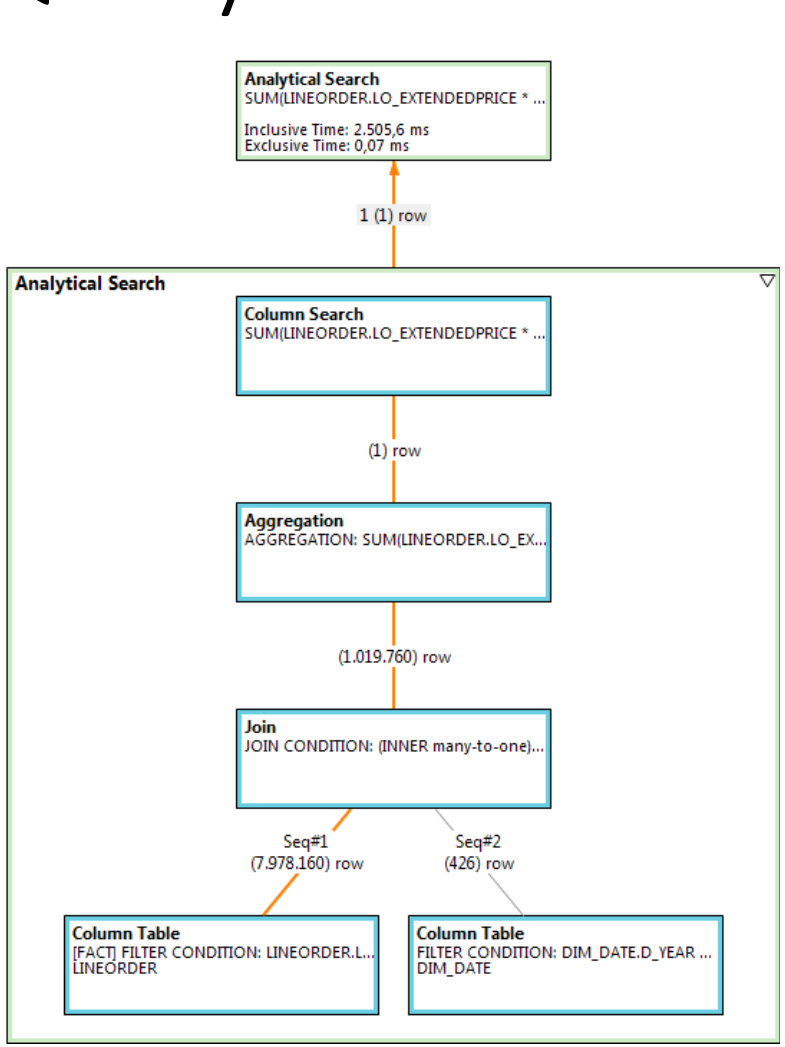
Row vs. Column Store

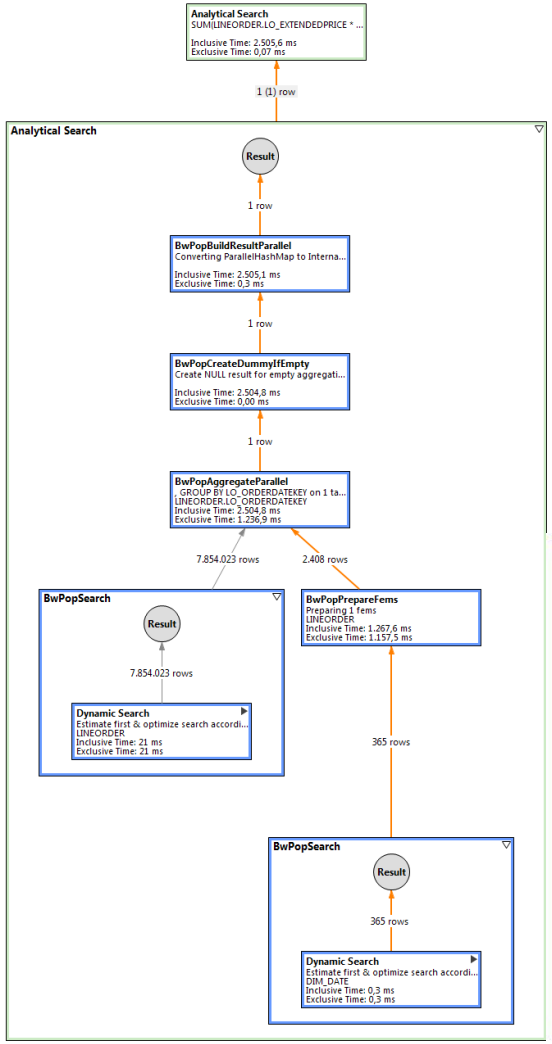
Indizes

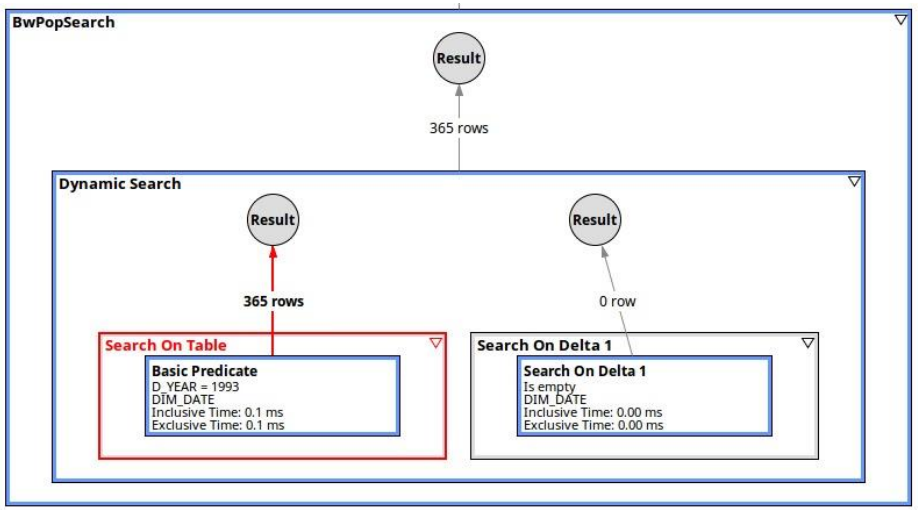
Hints

Foreign Key (FK)	+ Faktentabelle (FT)	+ Nur Dimensionen (DimOnly)	+ Restriktive Indizes auf Dimensionen (RestrDim)	Keine Indizes (None)
lo_custkey	lo_quantity	c_region	c_city	
lo_suppkey	lo_extendedprice	c_mktsegment	p_brand	
lo_partkey	lo_discount	p_mfgr	s_city	
lo_orderdatekey		p_category	d_yearmonthnum	
lo_commitdatekey		s_nation	d_yearmonth	
		s_region	d_daynuminyear	
		d_year		

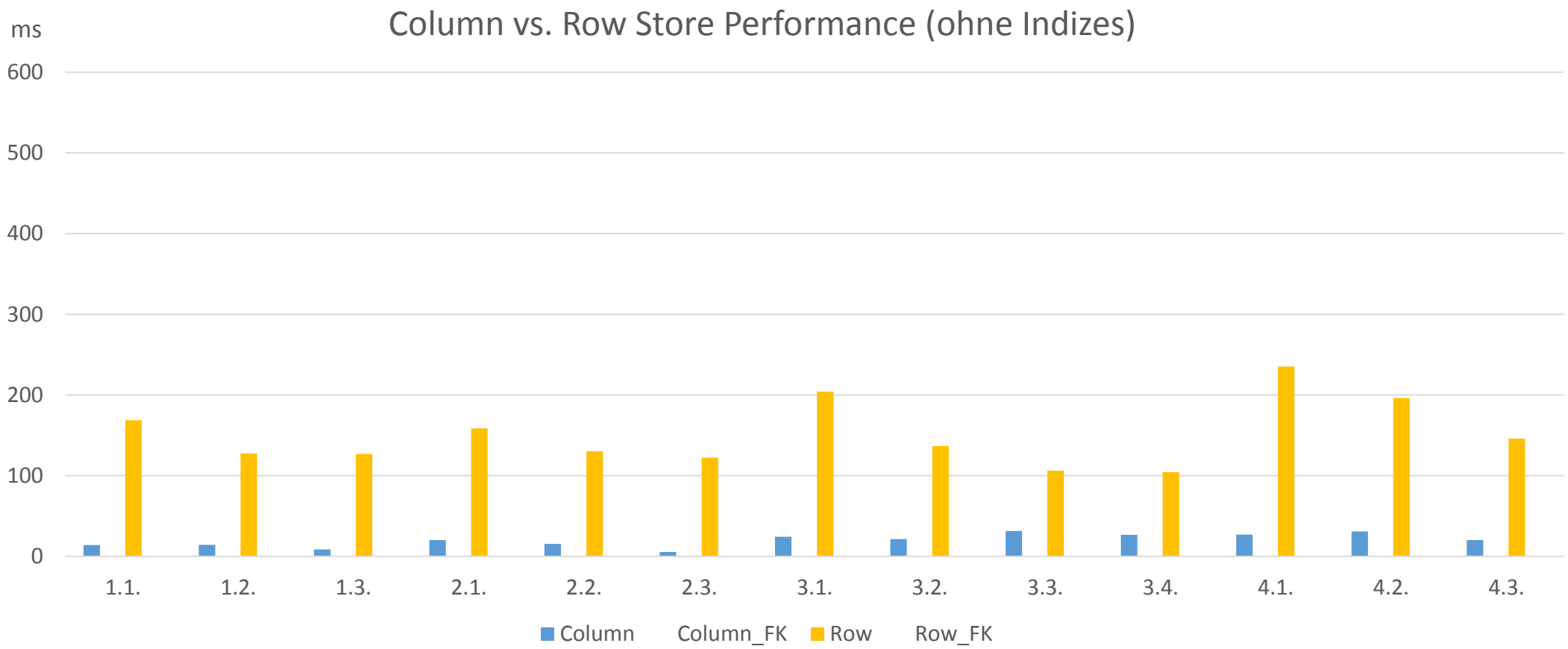
Query Execution Plans



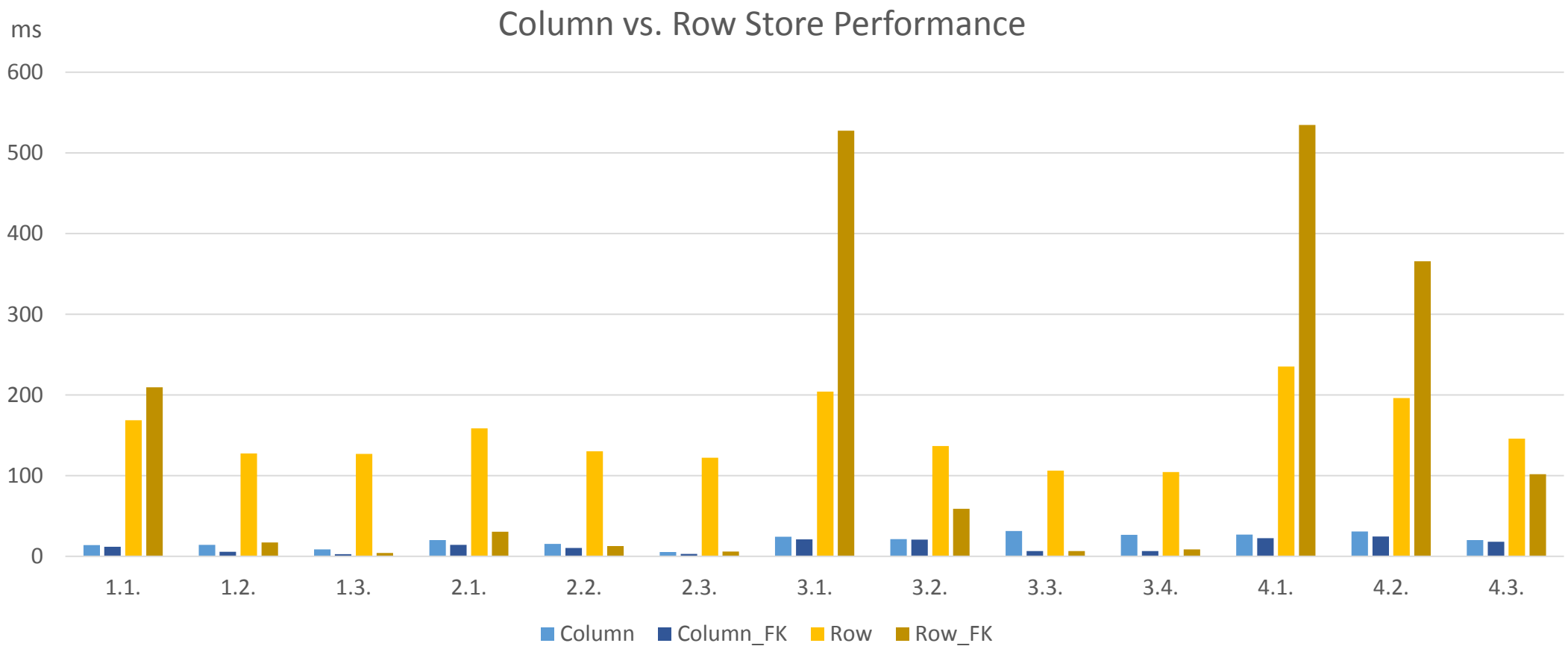




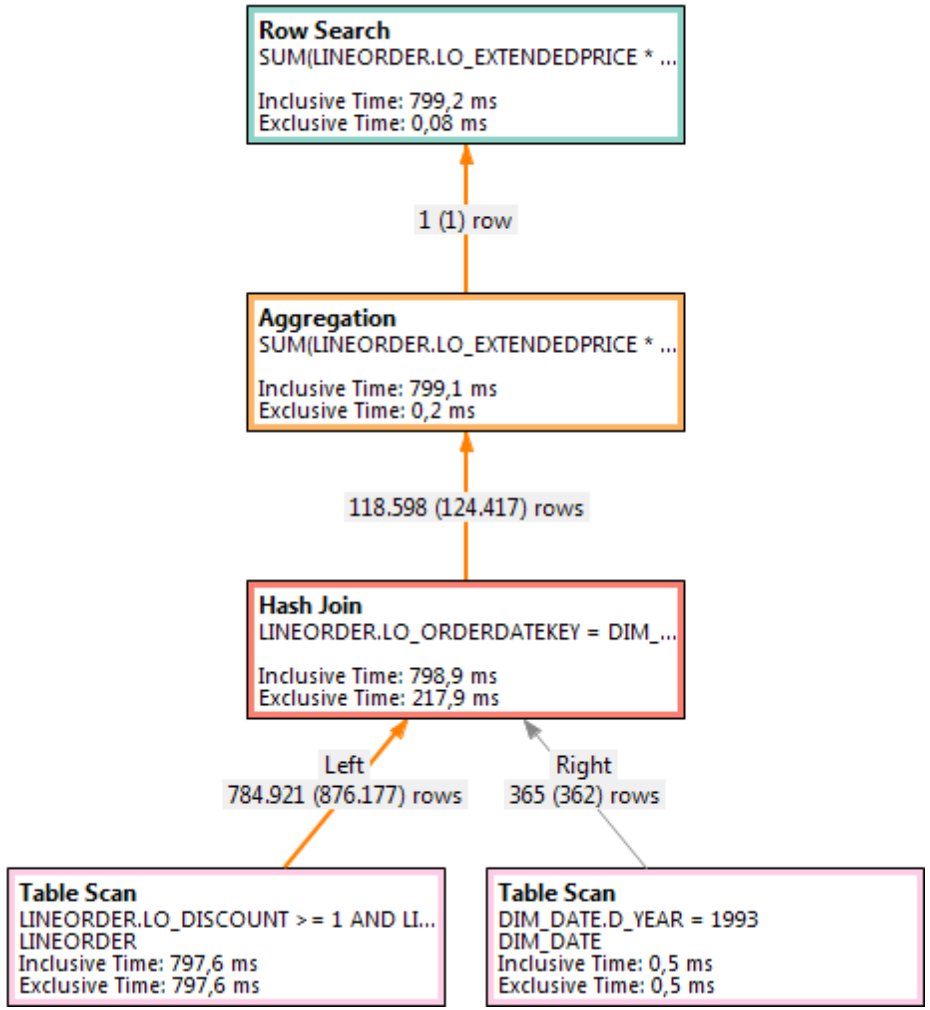
Column vs. Row Store

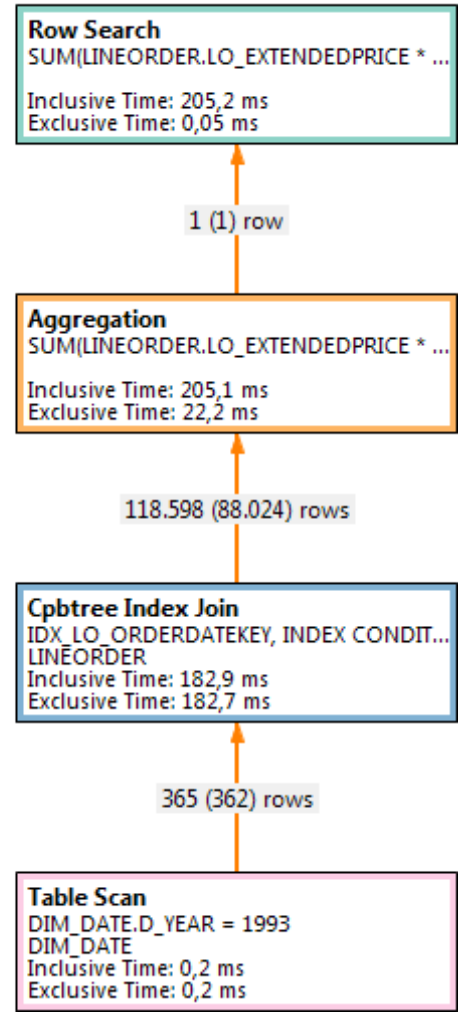


Column vs. Row Store

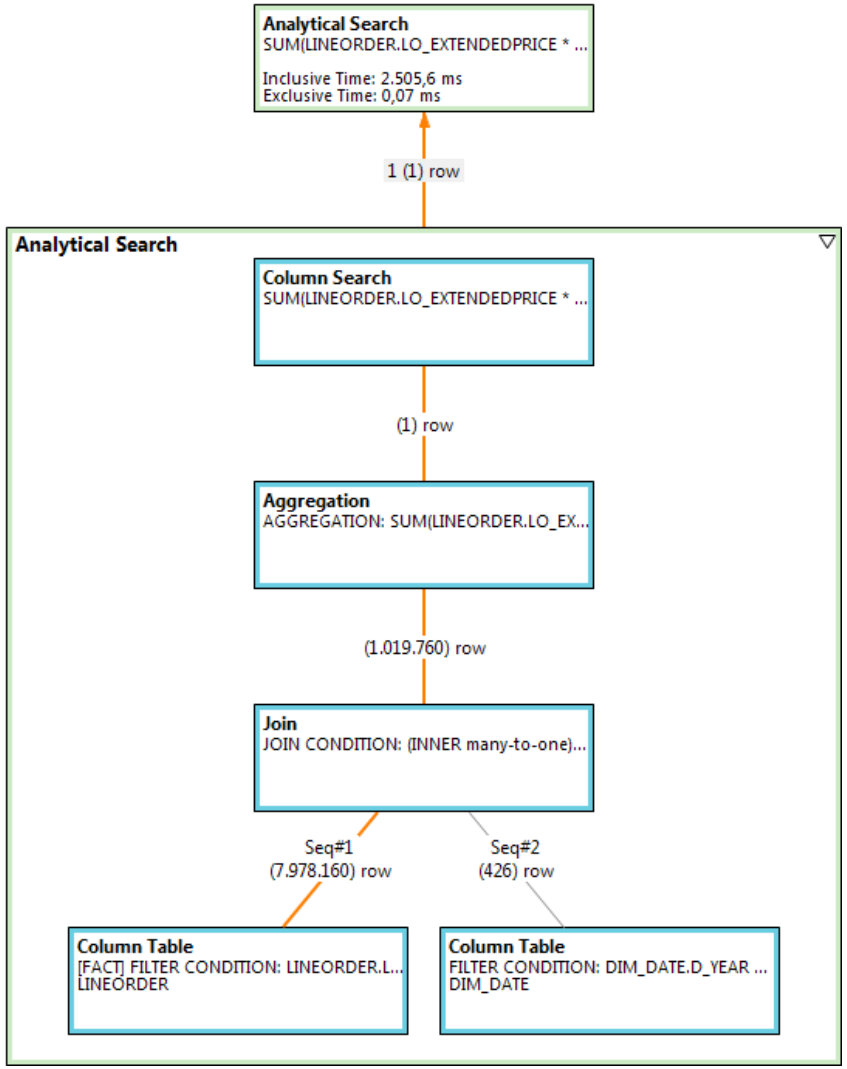


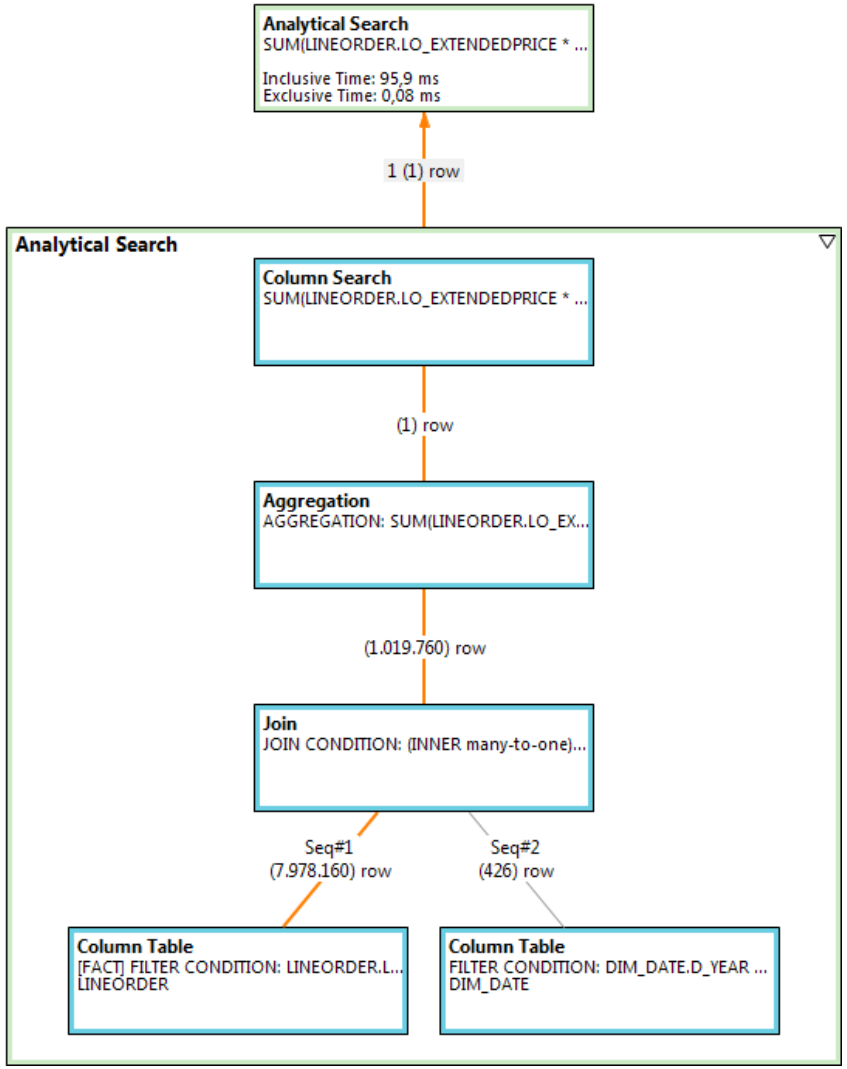
Row Store mit und ohne Indizies



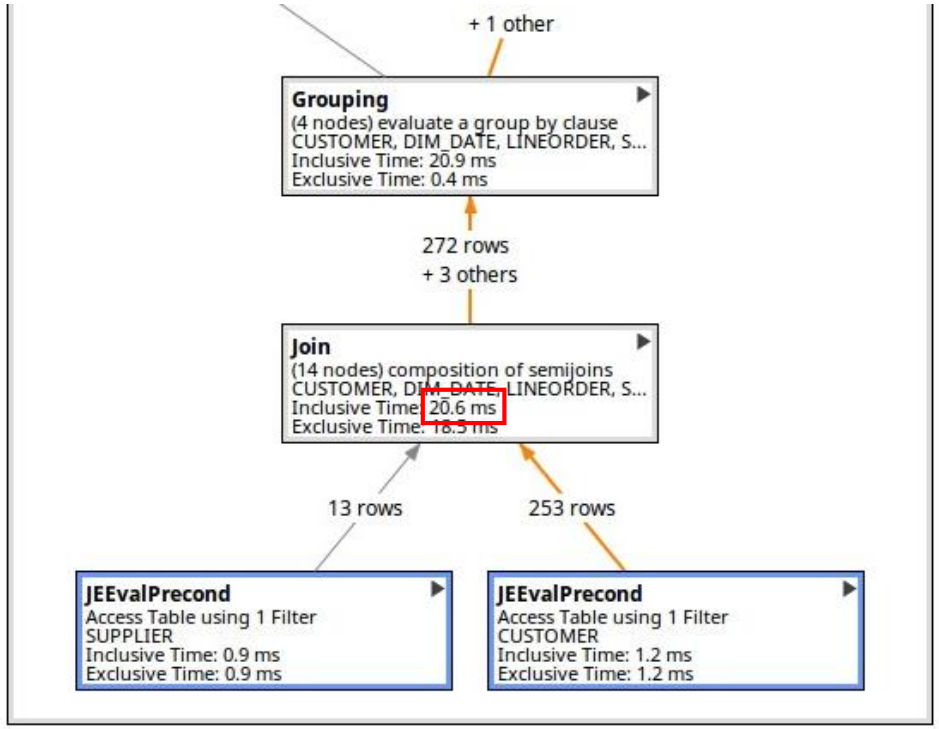


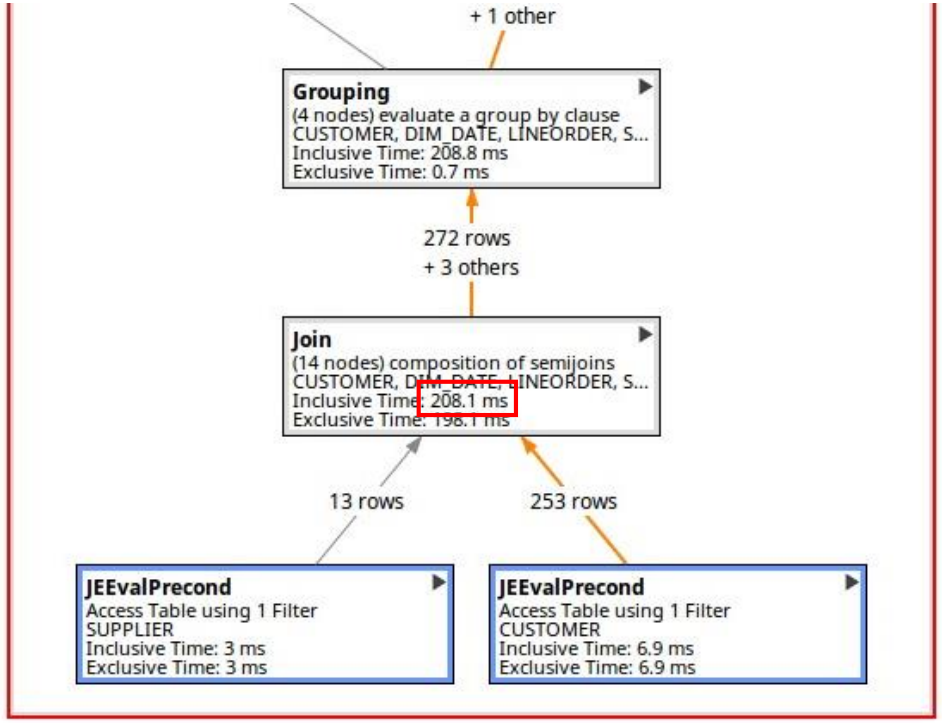
Column Store ohne und mit Indizes



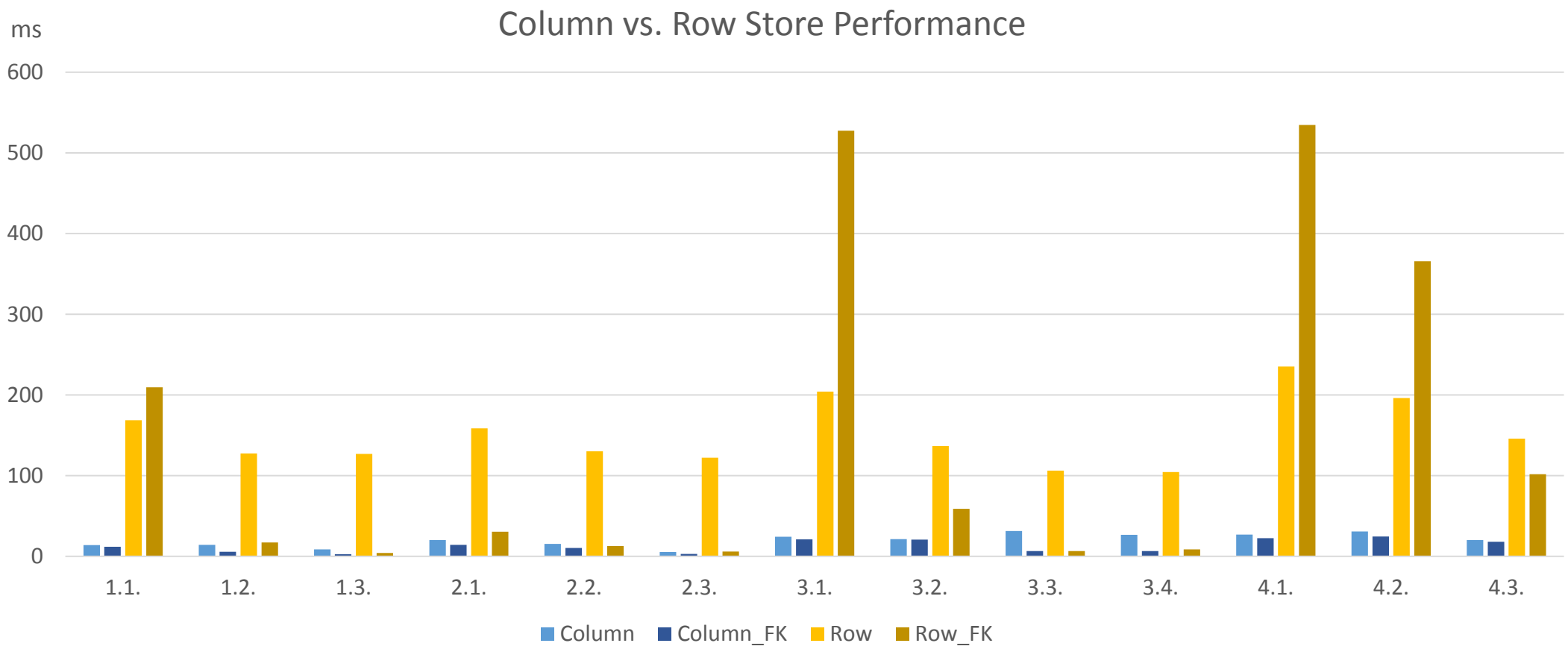


Column Store mit und ohne Indizies (3.3)

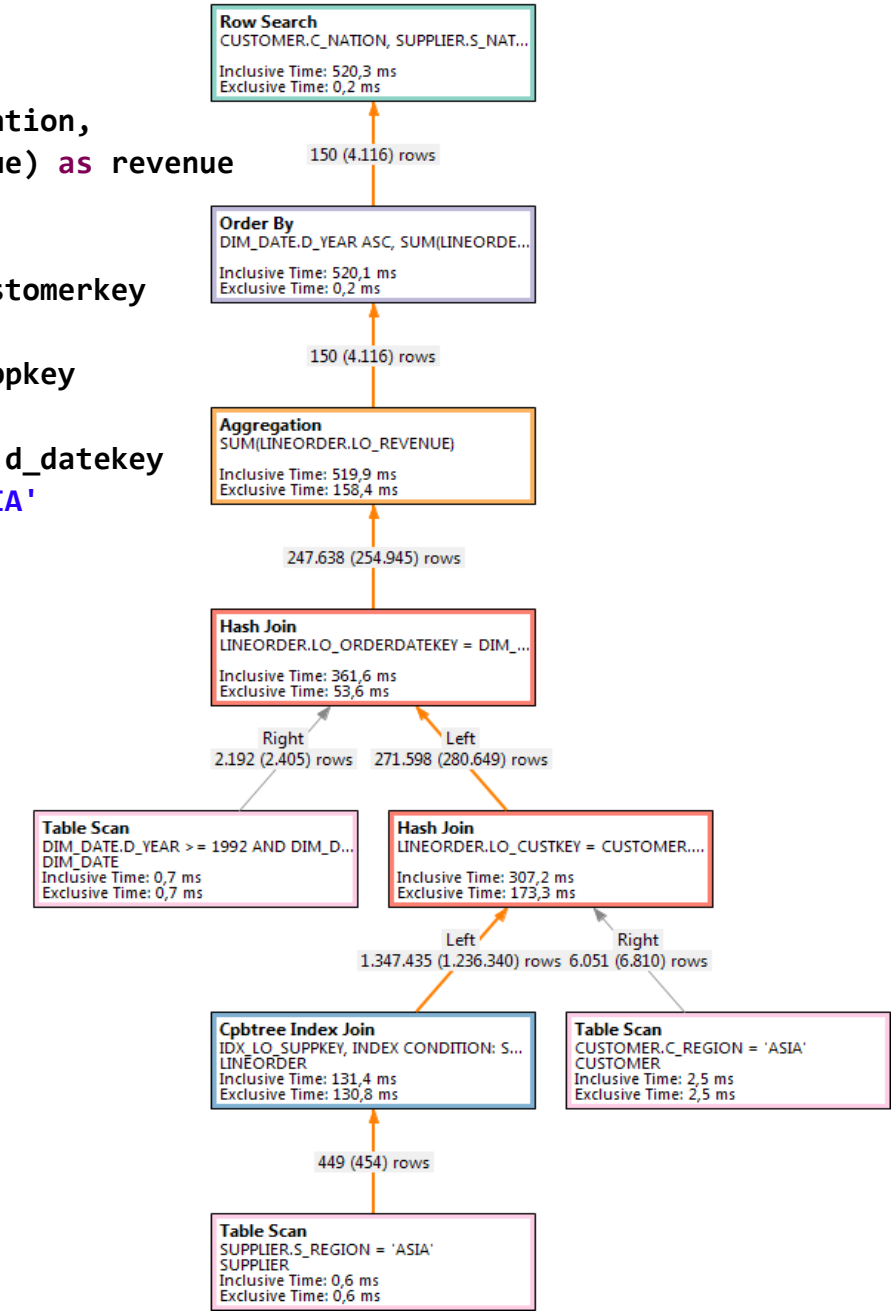


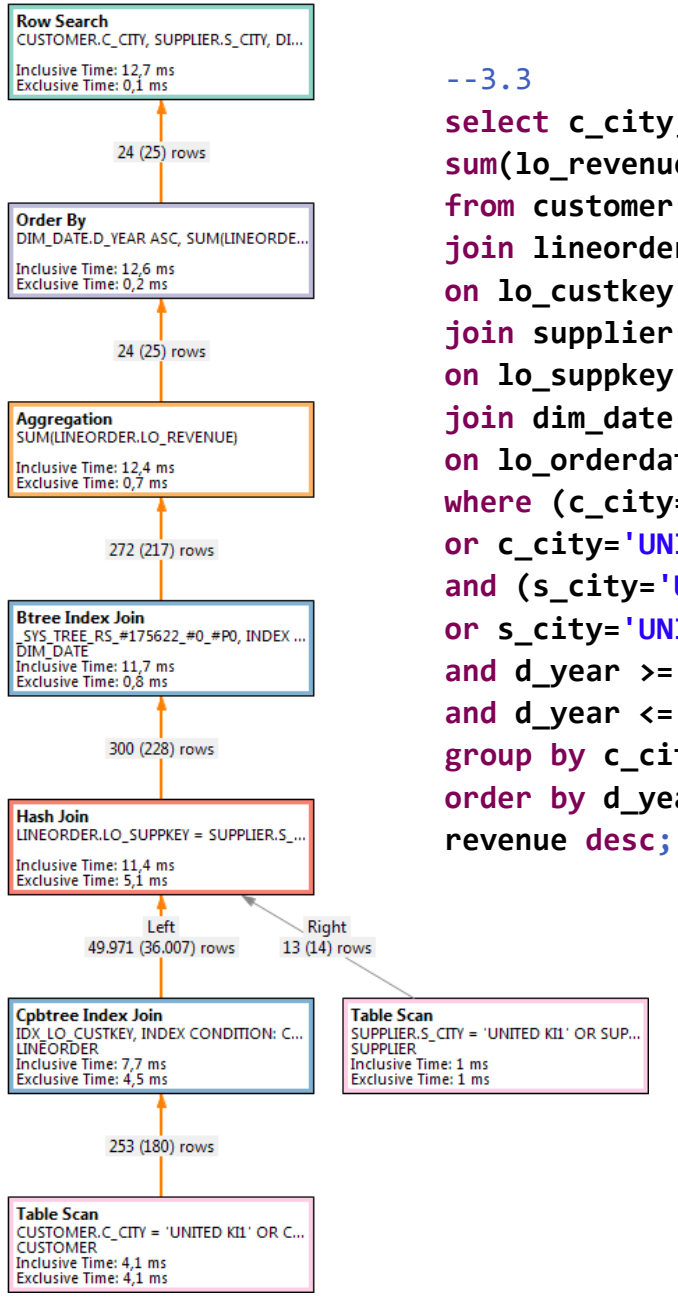


Column vs. Row Store



```
--3.1
select c_nation, s_nation,
d_year, sum(lo_revenue) as revenue
from customer
join lineorder
on lo_custkey = c_customerkey
join supplier
on lo_suppkey = s_suppkey
join dim_date
on lo_orderdatekey = d_datekey
where c_region = 'ASIA'
and s_region = 'ASIA'
and d_year >= 1992
and d_year <= 1997
group by c_nation,
s_nation, d_year
order by d_year asc,
revenue desc;
```



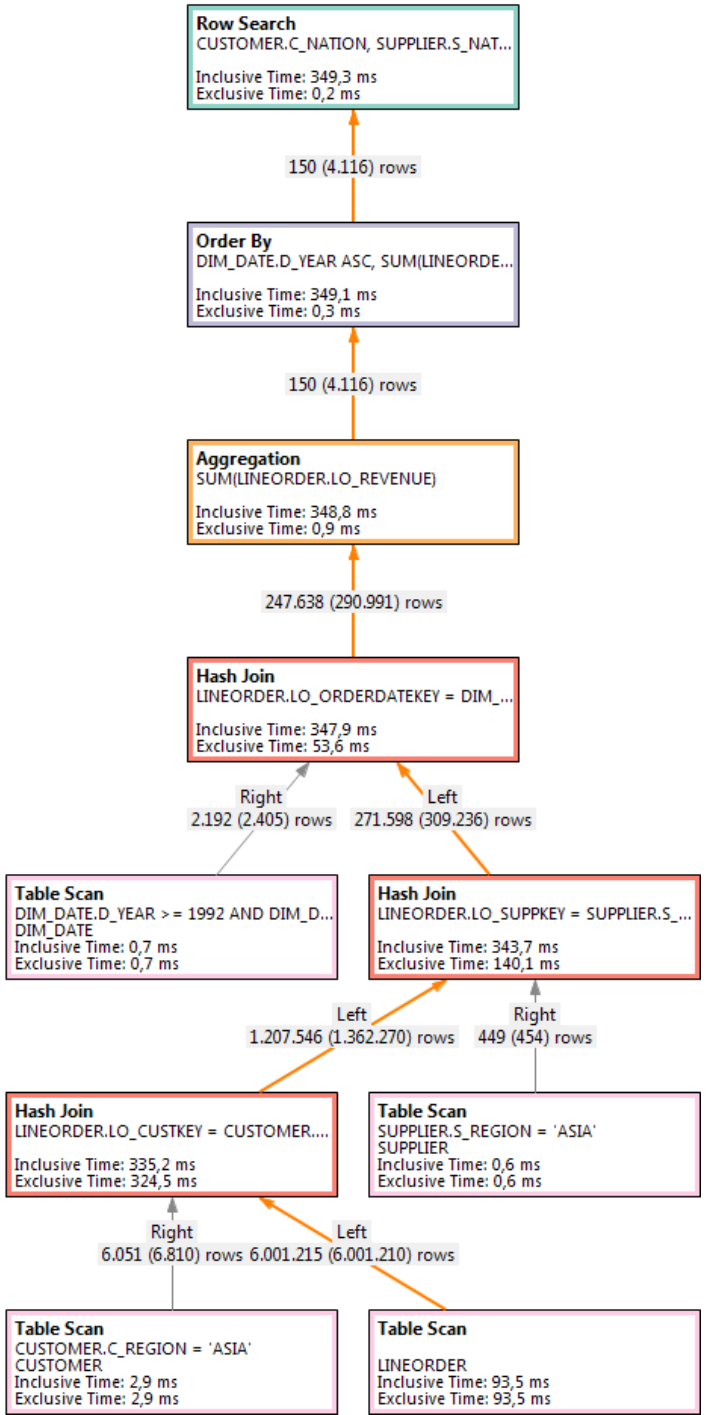



```
--3.3
select c_city, s_city, d_year,
sum(lo_revenue) as revenue
from customer
join lineorder
on lo_custkey = c_customerkey
join supplier
on lo_suppkey = s_suppkey
join dim_date
on lo_orderdatekey = d_datekey
where (c_city='UNITED KI1'
or c_city='UNITED KI5')
and (s_city='UNITED KI1'
or s_city='UNITED KI5')
and d_year >= 1992
and d_year <= 1997
group by c_city, s_city, d_year
order by d_year asc,
revenue desc;
```

Q3.1

NO_INDEX_JOIN

1,653ms => 1,323ms



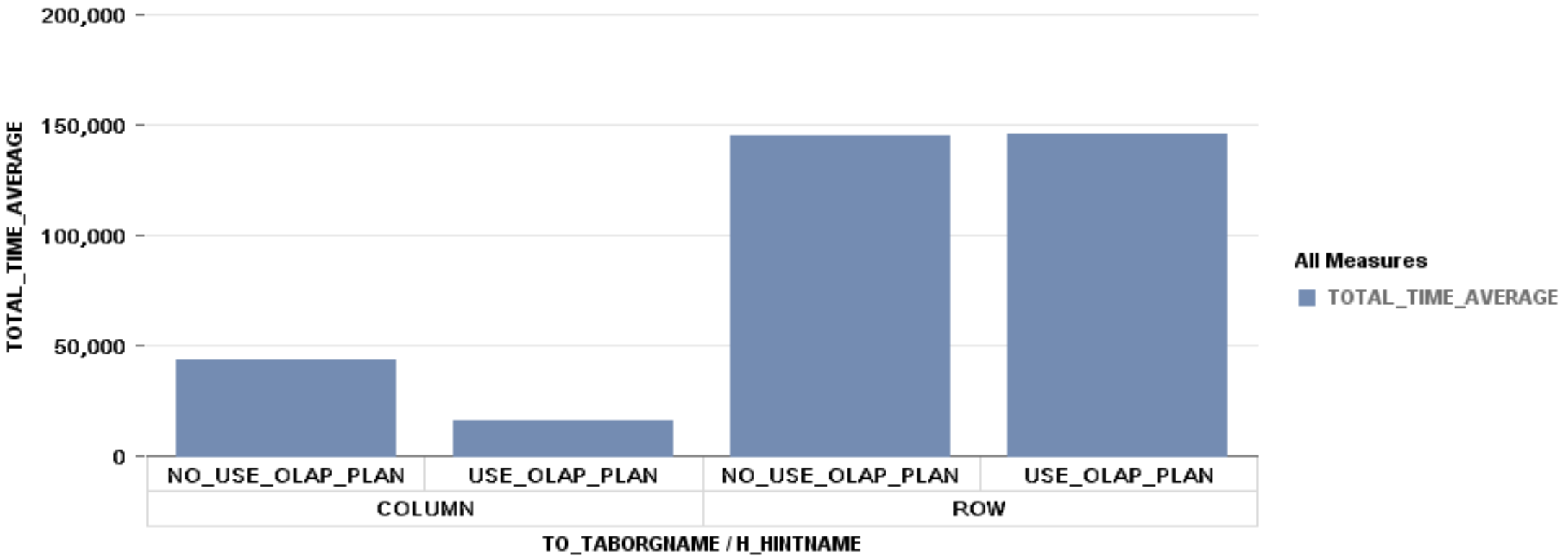
NO_HASH_JOIN => 3,609ms

RANGE_JOIN => 1,696ms

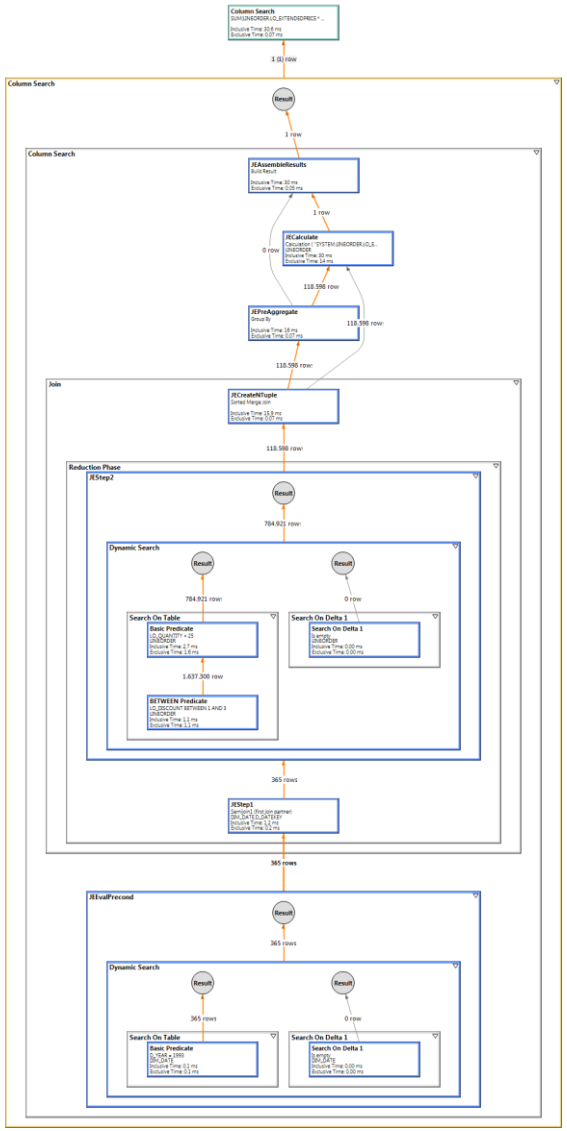
Column vs. Row Store: Indizes

	Column Benchmark	Column Benchmark with Indizes	Row Benchmark	Row Benchmark with Index
Samples	10	10	10	10
Total	5783	5182	20325	18850
Max	1718	559	2382	1934
Min	442	505	1913	1871
Median	453	517	2001	1880
Average	578	518	2033	1885
StandardDeviation	380	16	121	17

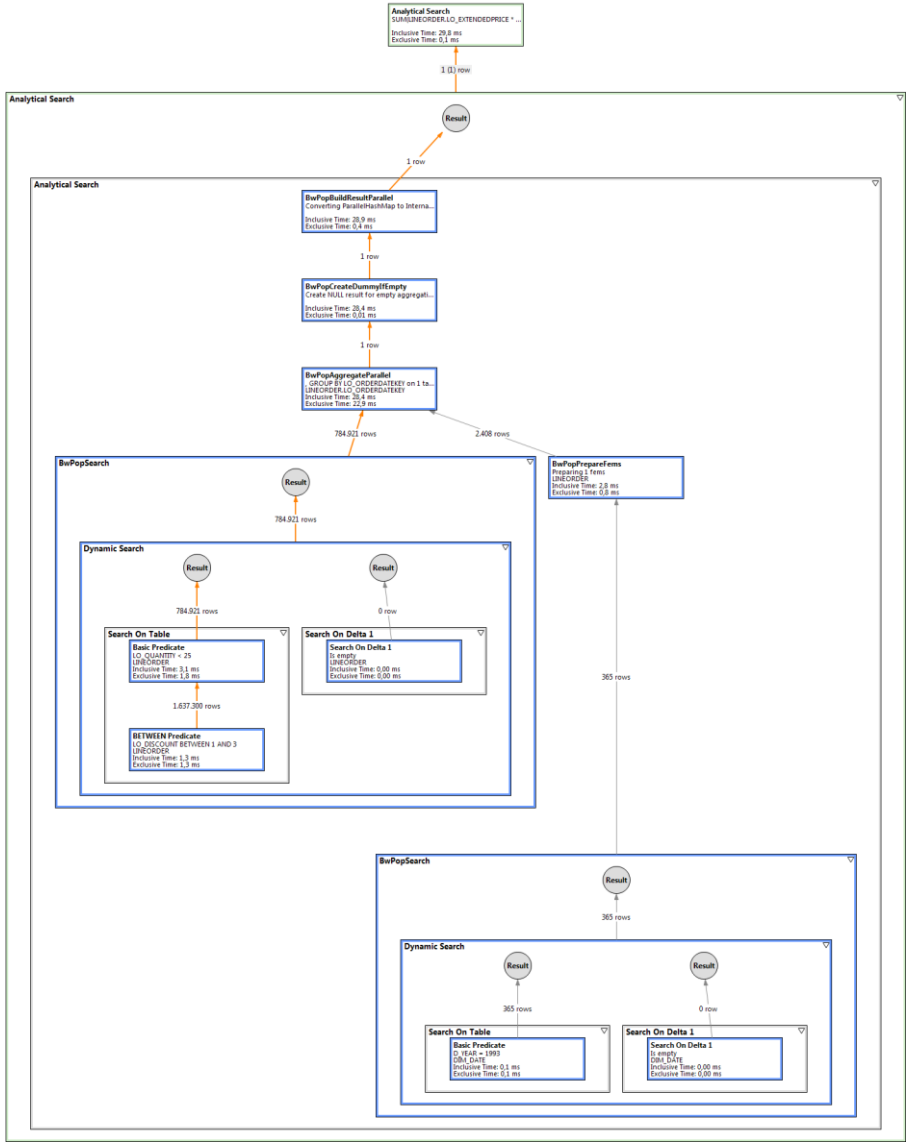
Column vs. Row Store

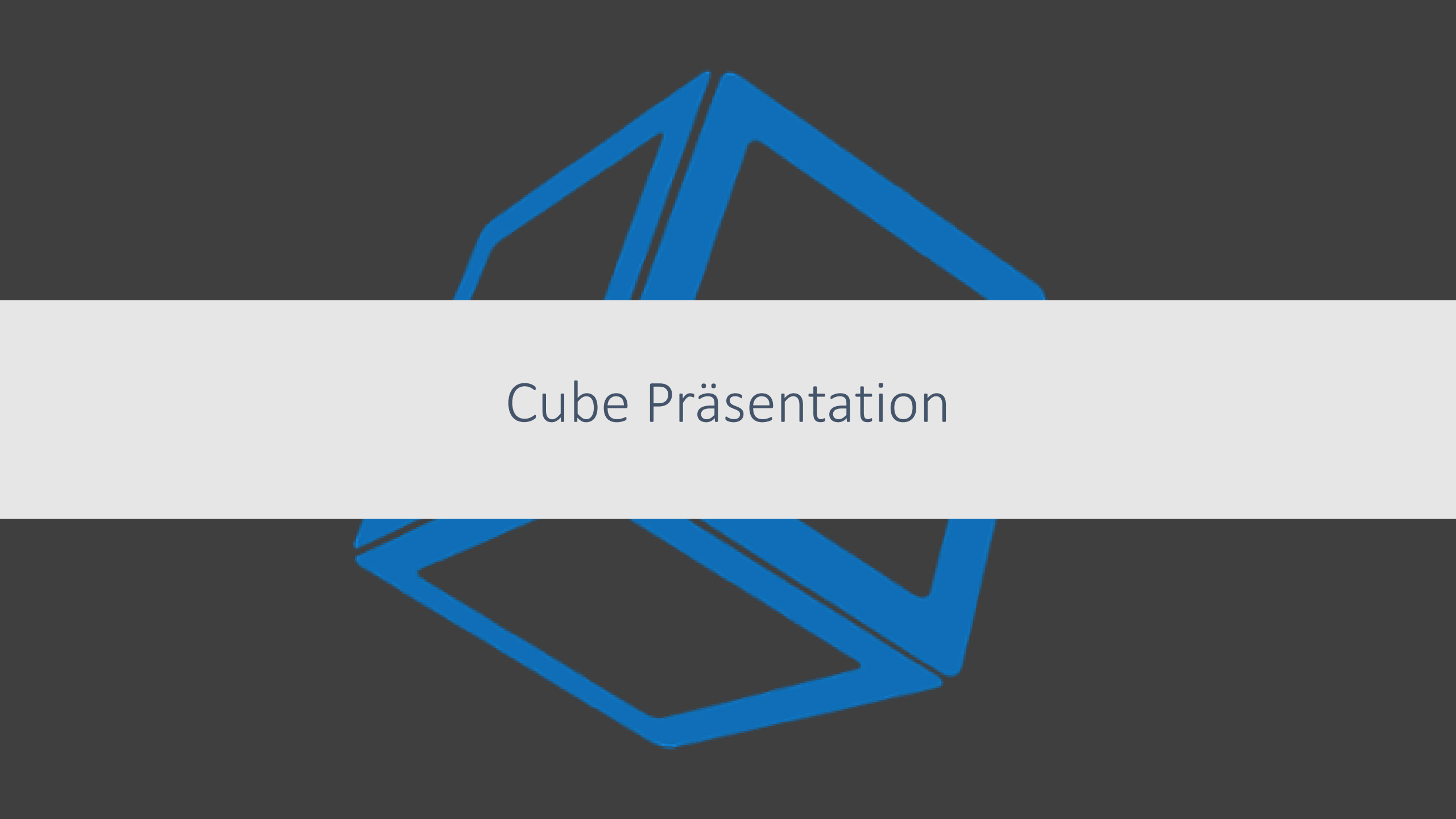


NO_USE_OLAP_PLAN



USE_OLAP_PLAN





Cube Präsentation

SSBM Benchmark auf HANA

- Column Store ist schneller
- Indizes können Column Store in manchen Fällen beschleunigen
- Der Optimizer entscheidet ob OLAP Plan verwendet wird
- HANA braucht viel RAM

22.03.2018

Kristina Albrecht, Jan Hofmeier, Lion Scherer, Marius Jochheim

SSBM Benchmark auf HANA

- Column Store ist schneller
- Indizes können Column Store in manchen Fällen beschleunigen
- Der Optimizer entscheidet ob OLAP Plan verwendet wird
- HANA braucht viel RAM

22.03.2018

DANKE!

Kristina Albrecht, Jan Hofmeier, Lion Scherer, Marius Jochheim