Databases for Massive Point Clouds

Massive Point Clouds for eSciences
http://pointclouds.nl

Peter van Oosterom (TUD), Oscar Martinez Rubi (NLeSc), Theo Tijssen (TUD), Mike Horhammer (Oracle) , Milena Ivanova (NLeSc), Romulo Goncalves (NLeSC)

SPAR Europe 2014, Amsterdam, The Netherlands
8-10 December 2014

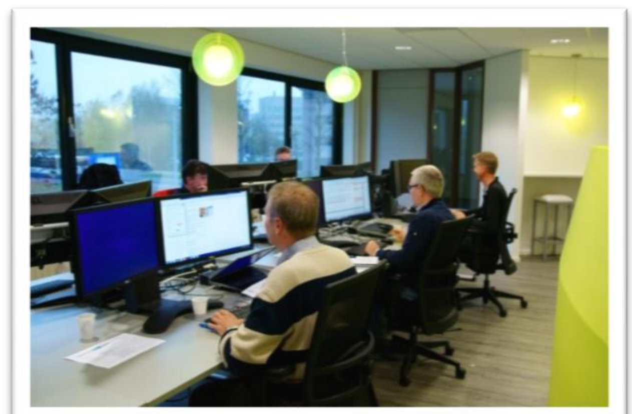
netherlands eScience center
by SURF & NWO

TU Delft Delft University of Technology

Netherlands eScience Center = *enhanced* Science

To reinforce and accelerate multi-disciplinary and data-intensive research in the Netherlands by developing and applying eScience and by combining forces.

enhanced Science is about promoting new scientific breakthroughs and innovation by bridging scientific disciplines via ICT

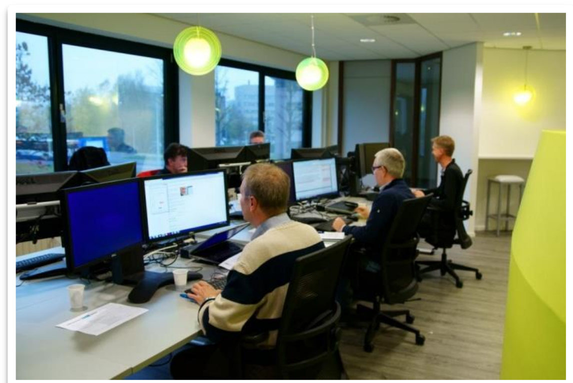


Optimizing Discovery in the Big Data Era

NLeSC: Innovation *with* ICT

Bridging the gap between Science and ICT

Bridging the gap between academic and commercial research



Content overview

Introduction

- Collection of point cloud data grows rapidly
  - Many new applications now economically viable
- Relevant use cases in GIS:
  - Digital elevation model of terrains
  - 3D models of urban environment
  - Flood risk management
  - Dike monitoring (NL)
  - Forest mapping
- Requiring new techniques for
  - Managing massive datasets
  - Integrating various kinds of data
- Other potential applications:
  - 3D models of manufactured parts
  - Visualization, animation, rendering
  - Medical imaging
  - Industrial metrology

Usually rasterized to be usable

Sitting on a gold mine, but not exploiting it!

Introduction

2 year Netherlands eScience research project on Massive Point Clouds

- TU Delft:
  - GIS technology
  - TU Delft, Library, contact with research & education users, dissemination & disclosure of point cloud data
  - 3TU.Datacentrum, Long-term provision of ICT-infra
  - TU Delft Shared Service Center ICT, storage facilities
- NL eScience Center, designing and building ICT infrastructure
- Oracle spatial, New England Development Centre (USA), improving existing software
- Rijkswaterstaat, data owner (and in-house applications)
- Fugro, point cloud data producer
- CWI, MonetDB group

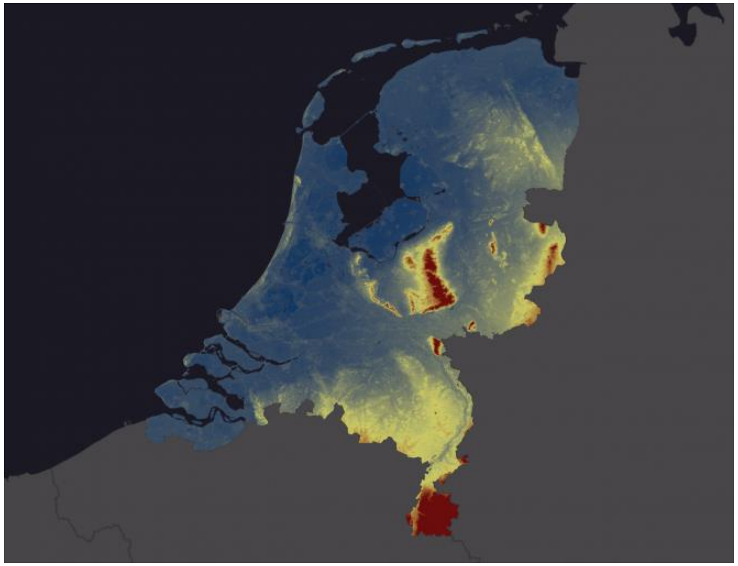


Image courtesy of: PDOK, NL

Introduction

Project motivation: Actual Height Model of the Netherlands (AHN2)

- Covering surface of the entire country
  - 6 -10 pts/m2 -> 640 billion pts
  - 60,185 LAZ files, **987 GB**/11.64 TB
  - (X, Y, Z) only
- “Future”
  - AHN3 higher resolution
  - Cyclorama-based photogrammetric datasets (50x AHN2, and with RGB)

Introduction

Project goals

- Develop infrastructure for the storage, the management, …
  of massive point clouds (note: no object reconstruction)
- Support range of hardware platforms: normal/ department servers (HP), cloud-based solution (MS Azure), Exadata (Oracle)
- Scalable solution: if data sets becomes 100 times larger and/or if we get 1000 times more users (queries), it should be possible to configure based on same architecture
- Generic, i.e. also support other (geo-)data and standards based, if non-existent, then propose new standard to ISO (TC211/OGC): Web Point Cloud Service (WPCS)
- Explore standardization at SQL level (SQL/SFS, SQL/raster, SQL/PC)?

Introduction

Evaluation of user requirements

- Based on structured interviews:
  - Government community: RWS (Ministry)
  - Commercial community: Fugro (company)
  - Scientific community: TU Delft Library
- Report at MPC public website http://pointclouds.nl
- Basis for conceptual benchmark:
  - Tests for functionality
  - Classified by importance

Introduction

Why a DBMS?

- Common practice: specific file format (LAS, LAZ, ZLAS,…) with specific tools / libraries
- Point clouds data similar to raster data:
  sampling nature, huge volumes, relatively static
- Missing features in specific file-based point cloud data management systems:
  - Multi-user (access and some update)
  - Scalability (not nice to process thousands/millions files)
  - Data integration (types: vector, raster, admin)
  - Online availability
- "work around" could be developed -> re-build DBMS
- No reason why point clouds can not be supported efficiently in DBMS
- Suggestion: "mix" both: use file (or GPU) format for the PC blocks

Content overview

1. Introduction
2. Conceptual benchmark
3. Executable benchmark
4. Data organization
5. Conclusion

Conceptual benchmark

SQL query types / functionality

1. simple range/rectangle filters (of various sizes) -> 10
2. selections based on points along a linear route (with buffer) -> 8
3. selections of points overlapping a 2D polygon -> 9
4. selections based on the attributes such as intensity I (/RGB) -> 8
5. multi-resolution/LoD selection (select top x%) -> 8, compute imp
6. sort points on relevance/importance (support streaming) -> 7
7. slope orientation or steepness computation -> 3

⋮

32. delta selection of query 31, moving to new position -> 6

Conceptual benchmark

Benchmark organization

- mini-benchmark, small subset of data

  (20 million = 20.000.000) + limited functionality
  - get experience with benchmarking, platforms
  - first setting for tuning parameters: block size, compression.
- medium-benchmark, various subsets with different sizes

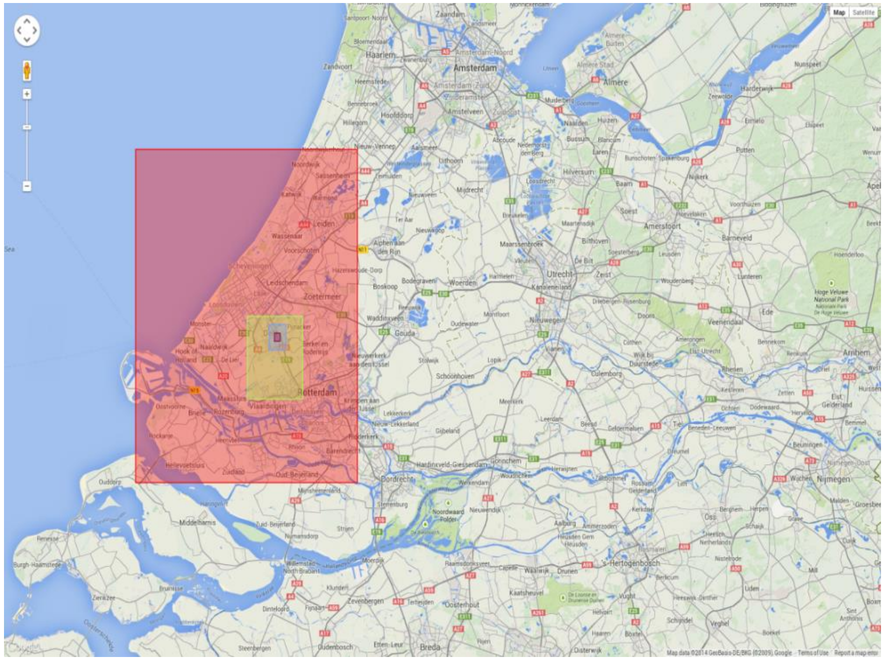
  (up to 20 billion = 20.000.000.000) + more functionality
  - more serious testing, first feeling for scalability
  - more and different types of queries (e.g. nearest neighbour)
- full-benchmark, full AHN2 data set

  (640 billion = 640.000.000.000) + yet more functionality
  - LoD (multi-scale), multi-user test
- scaled-up benchmark, replicated data set

  (20 trillion = 20.000.000.000.000)
  - stress test

Conceptual benchmark

Tested data: AHN2 (subsets)



Dataset name	Benchmark	Points	Files	Format	Disk size [GB]	Area [km²]	Description
20M	Mini/Medium	20,165,862	1	LAS	0.4	1.25	TU Delft campus
210M	Medium	210,631,597	16	LAS	4.0	11.25	Major part of Delft city
2201M	Medium	2,201,135,689	153	LAS	42.0	125	City of Delft and surroundings
23090M	Medium	23,090,482,455	1,492	LAS	440.4	2,000	Major part of Zuid-Holland province
639478M	Full	639,478,217,460	60,185	LAZ	987.0*	40,000	The Netherlands

HP DL380p Gen8 server
“normal” server hardware configuration

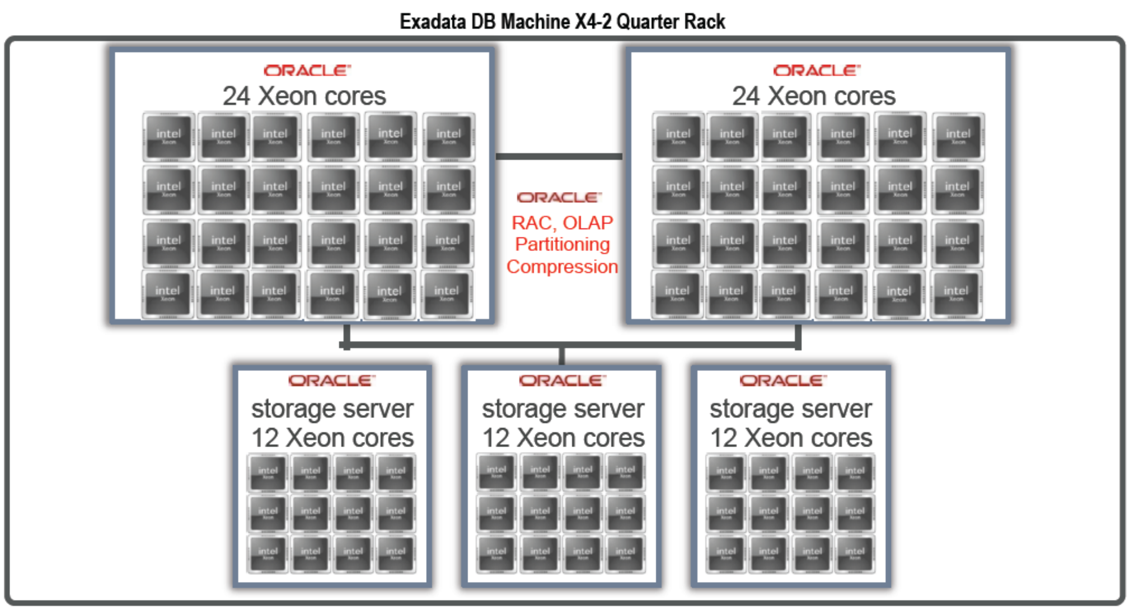
- HP DL380p Gen8 server
  - 2 x 8-core Intel Xeon processors, E5-2690 at 2.9 GHz
  - 128 GB main memory (DDR3)
  - RHEL 6.5 operating system

- Disk storage – direct attached
  - 400 GB SSD (internal)
  - 6 TB SAS 15K rpm in RAID 5 configuration (internal)
  - 2 x 41 TB SATA 7200 rpm in RAID-5 configuration (external in 4U rack 'Yotta-III' box, 24 disks)

Exadata X4-2

Oracle SUN hardware for Oracle database software

- Database Grid: multiple Intel cores, computations
  Eight, quarter, half, full rack with resp. 24, 48, 96, 192 cores
- Storage servers: multiple Intel cores, massive parallel smart scans (predicate filtering, less data transfer, better performance)
- Hybrid columnar compression (HCC): query and archive modes



"DBMS counterpart of GPU for graphics"

Content overview

1. Introduction
2. Conceptual benchmark
3. Executable benchmark
4. Data organization
5. Conclusion

Executable benchmark

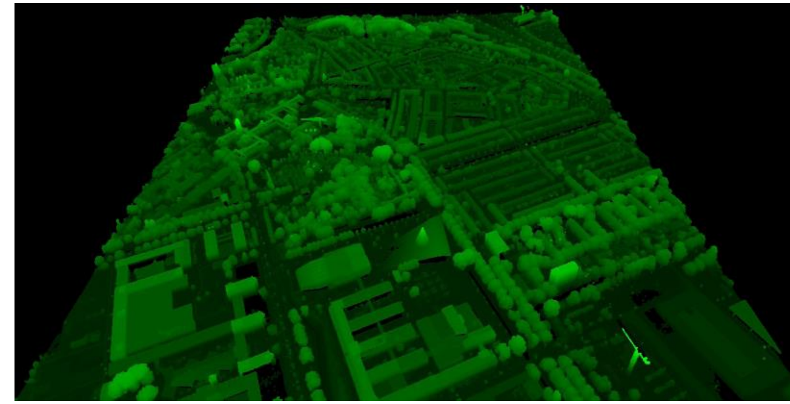
Point cloud data management systems

- Oracle
  - Blocks
    (native PointCloud support)
  - Flat table
- PostgreSQL
  - Blocks
    (PointCloud extension)
  - Flat table
- LASTools
- MonetDB

  Flat table (column-store)

Executable benchmark

mini-benchmark



- Load small AHN2 dataset 20M (20.165.862 XYZ points)
- X, Y, Z
- Management systems:
  - PointCloud (blocks) solutions in Oracle and PostgreSQL

    no compression, block size 5000, one thread
  - Flat tables: Oracle, PostgreSQL and MonetDB

    1 point (x,y,z) per row.
    - Oracle: Btree
    - PostgreSQL: GiST
    - MonetDB: Imprints
  - LAStools (file, no database, tools from rapidlasso, Martin Isenburg)
- 7 queries

Executable benchmark

Additional initial tests. Example: block sizes and compression

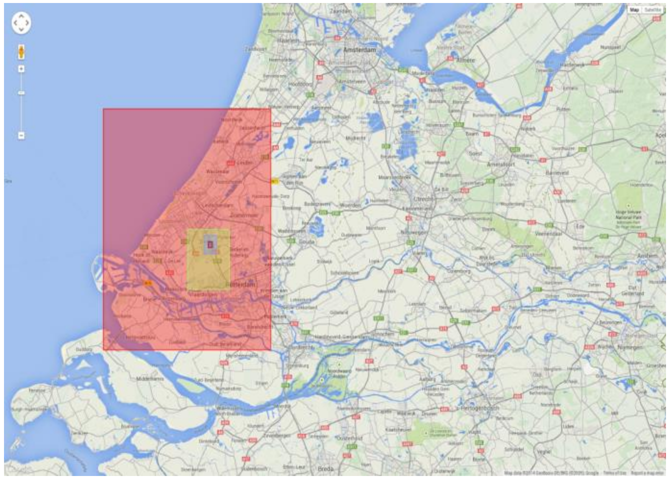
- Block size: 300, 500, 1000, 3000 and 5000 points
- Compression:
  - Oracle blocks: none, medium and high
  - PostgreSQL blocks: none, dimensional

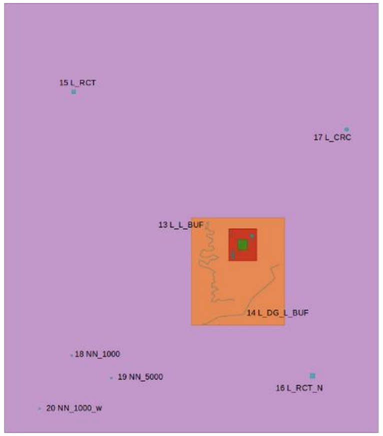
- Conclusions (most the same for PostgreSQL, Oracle):
  - Compression about factor 2 to 3 (not as good as LAZ/ZLAS: 10)
  - Load time and storage size are linear to size datasets
  - Query time not much different: data size / compression (max 10%)
  - Oracle medium and high compression score equal
  - Oracle load gets slow for small block size 300-500

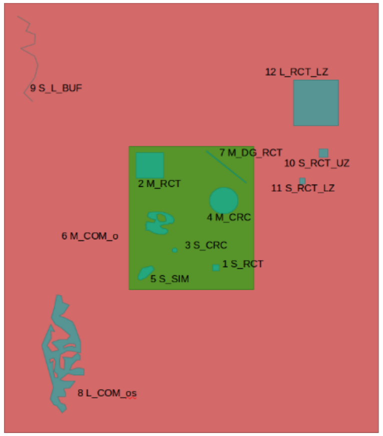
Executable benchmark

Medium benchmark

- Four datasets: 20M, 210M, 2201M, 23090M
- X, Y, Z
- Best known loading strategies (parallel out-core)
- Best known configuration (block sizes)
- Use compression
- LAStools needs help when scaling
- Compare with Exadata (diff. Hardware)
- 20 queries (parallel, out-core when not native)

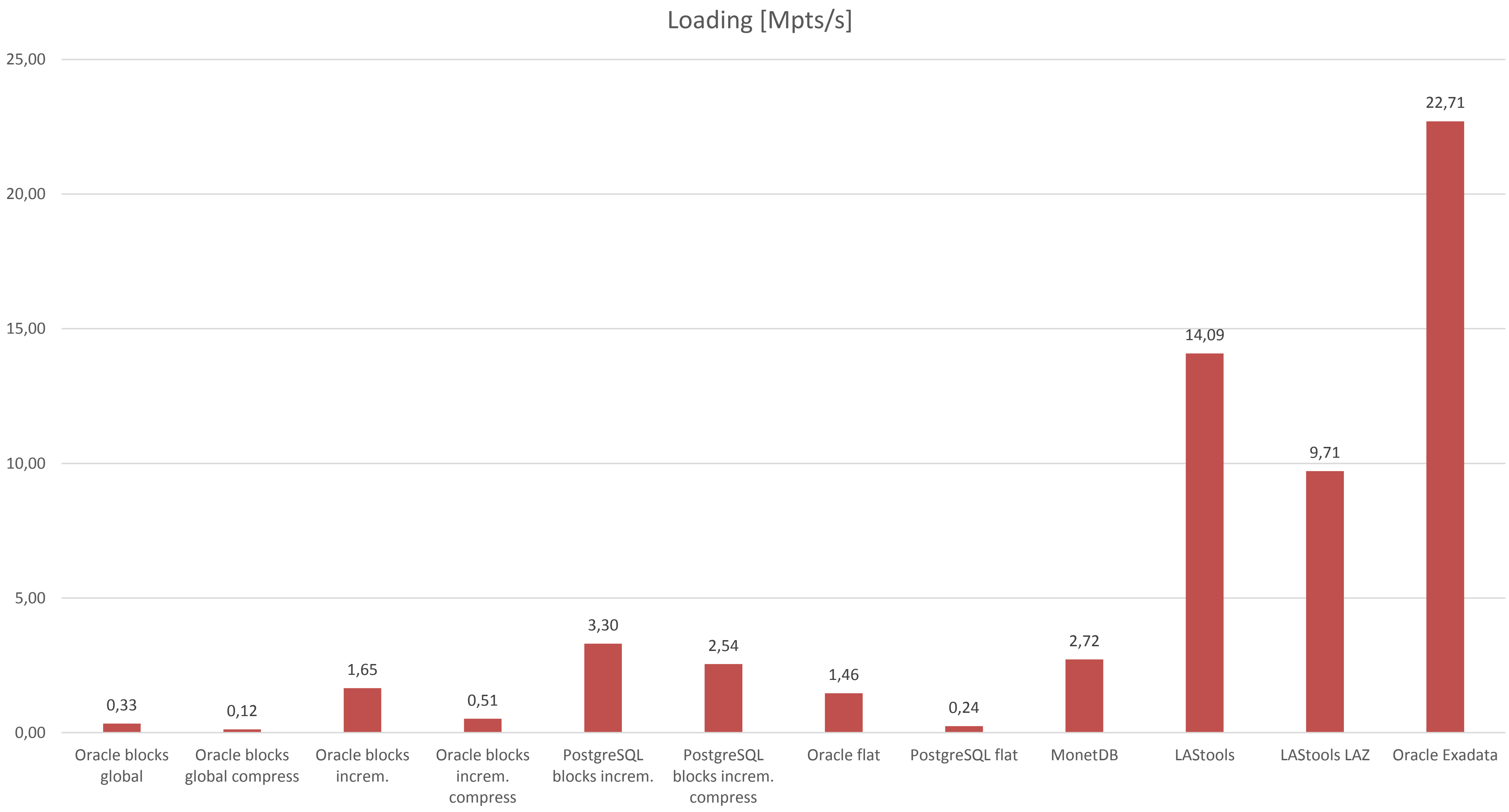






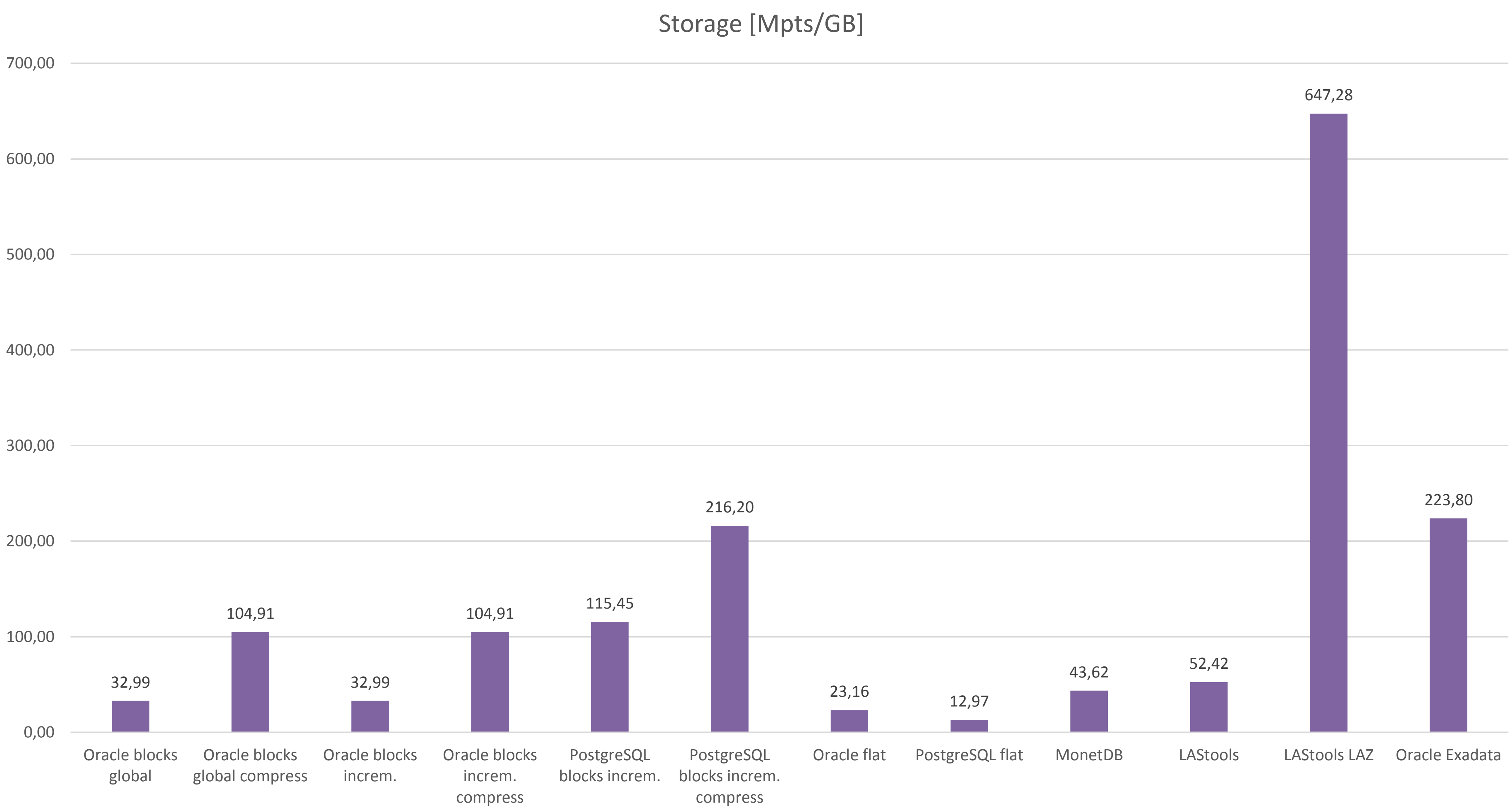
Executable benchmark

Medium benchmark: Loading of 23090M points



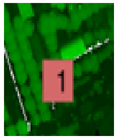
Executable benchmark

Medium benchmark: Storage of 23090M points

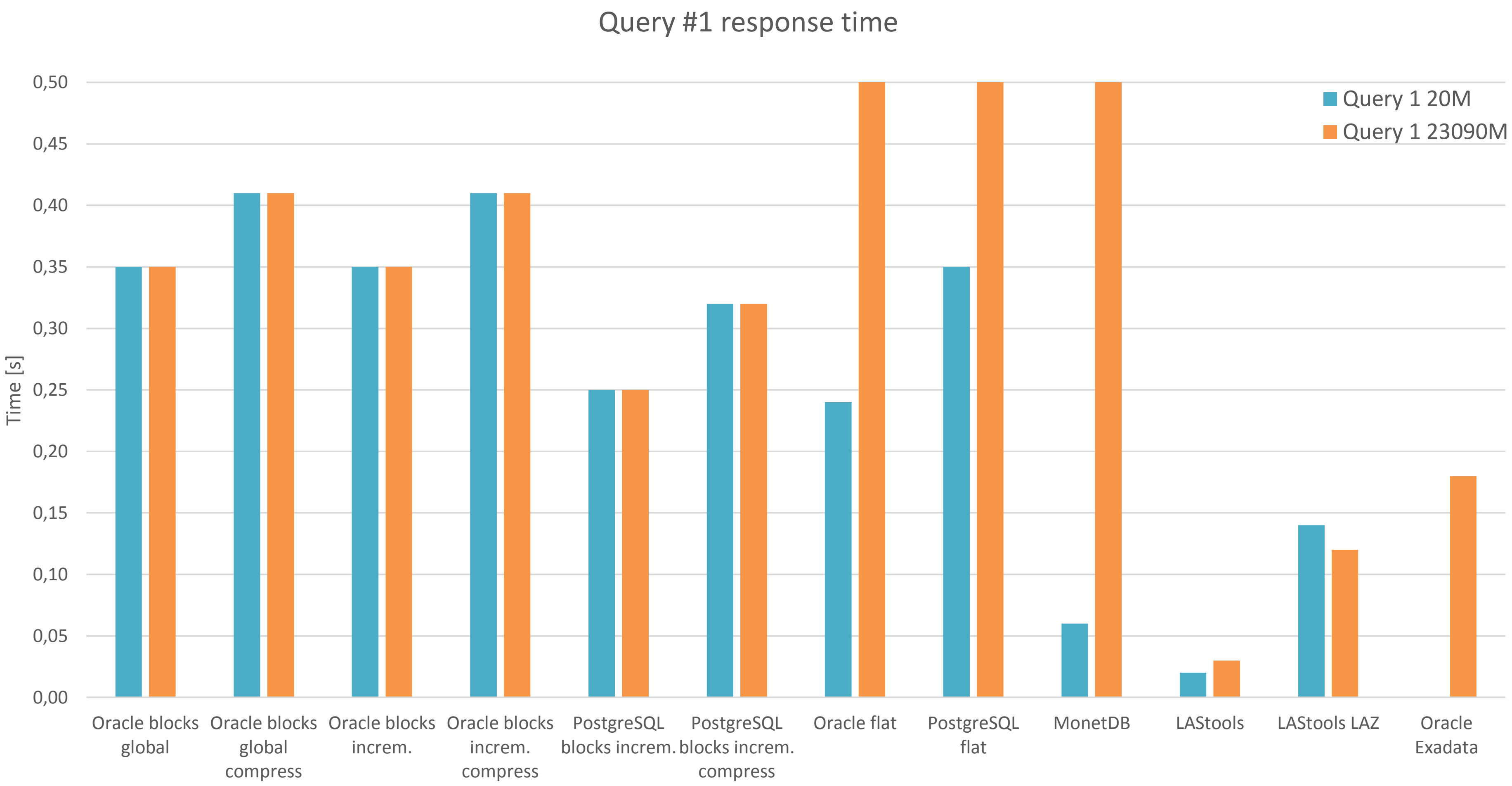


Executable benchmark

Medium benchmark: Queries - Q1 (20M and 23090M)

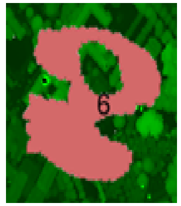


Rectangle 51x53 m, 0.0027 km^2, ~74872 points

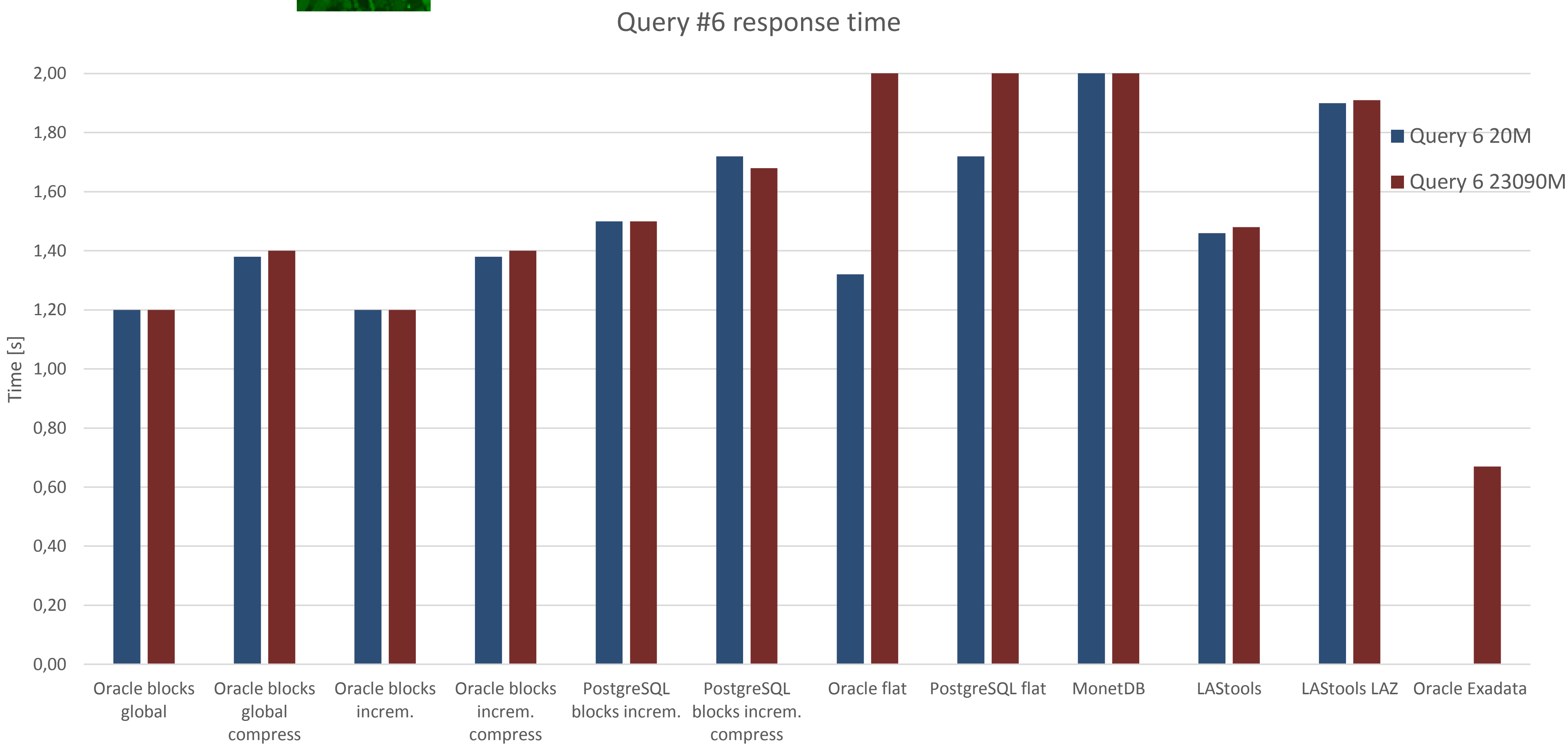


Executable benchmark

Medium benchmark: Queries – Q6 (20M and 23090M)



Complex Polygon, 792 pts, 0.025 km², 1 hole, ~387135 points



netherlands eScience center

TU Delft Delft University of Technology

Executable benchmark

Full benchmark: loading AHN2

system	Total load time [hours]	Total size [TB]	#points
LAStools unlic.	22:54	12.181	638,609,393,087
LAStools lic	16:47	11.617	638,609,393,101
LAStools lic LAZ	15:48	0.973	638,609,393,101
Oracle Exadata	4:39	2.240	639,478,217,460
MonetDB*	17:21	15.00	639,478,217,460

Executable benchmark

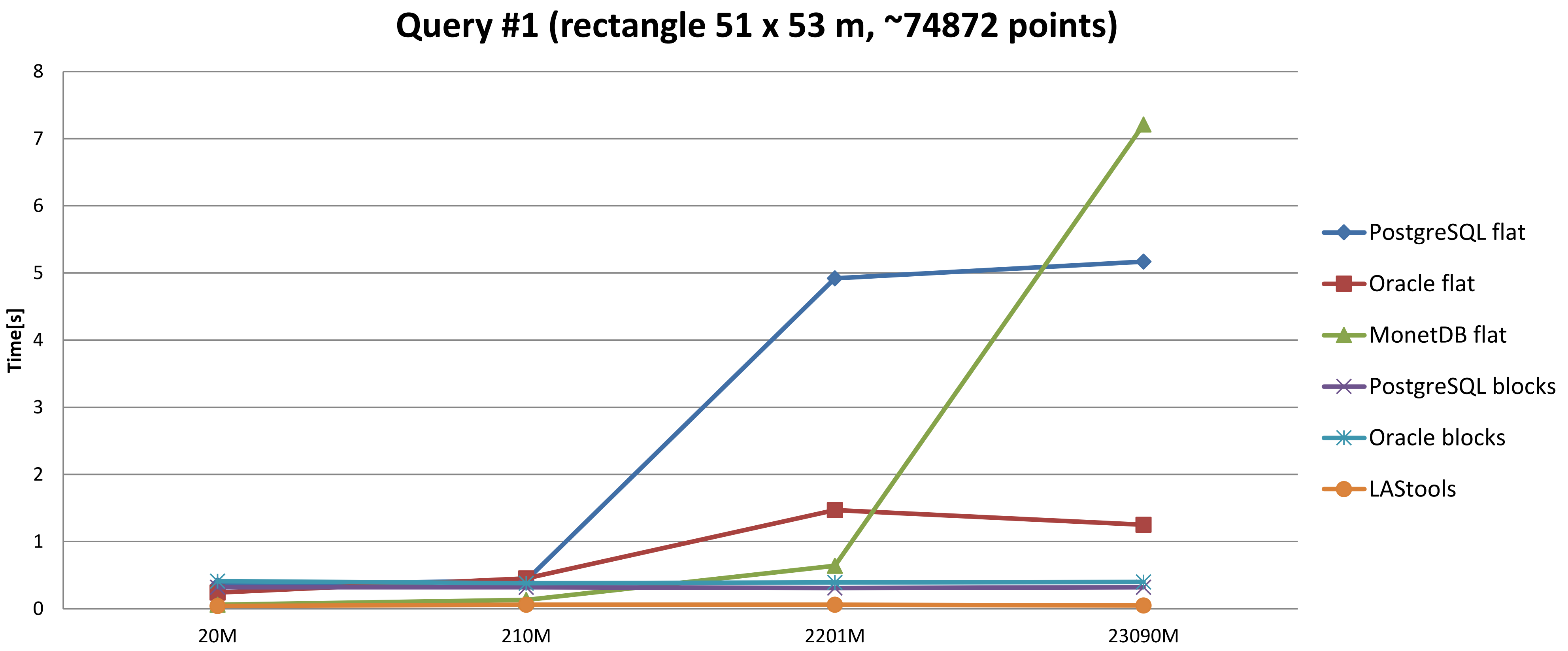
Storage model	Pro	Con
DB blocks	• Storage (compression) • Scaling • Indexing • DB functionalities • Complex queries	• Loading (make blocks) • Block overhead in queries (noticeable in simple queries) • Not native parallel
DB flat	• Faster loading/updating* • DB functionalities • Dynamic schema • Simple queries • Native parallelization (not PostgreSQL)	• Storage (except Exadata) • Not scaling (except Exadata) • Indexing (except Exadata)
File-based	• Storage (LAZ) • Data preparation • Simple queries (if not LAZ)	• Limited functionalities • Fixed schema (LAS) • Scaling requires DB help • Not efficient parallel

Content overview

1. Introduction
2. Conceptual benchmark
3. Executable benchmark
4. Data organization
5. Conclusion

Data organization

Querying current situation



Data organization

Querying current situation

- Flat tables fast when small dataset but not scaling
- Blocks not so fast but “perfectly” scaling (better storage but need “blocks” processing in queries)
- What if we want DBMS with speed of small flat tables but good scaling?
  - How can a flat table be organized efficiently?
  - How can the point cloud blocks be created efficiently?
    (with no assumption on data organization in input)
  - ANSWER: spatial clustering/coherence, e.g. quadtree/octree
    (as obtained by Morton or Hilbert space filling curves)

Data organization

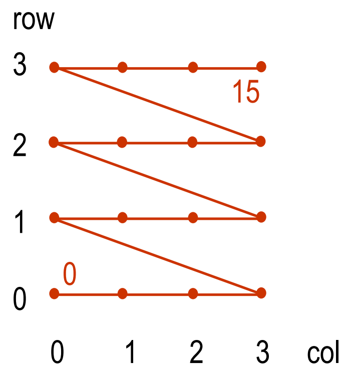
Spatial clustering

- Reorder data based on spatial dimensions
  - Efficiency in storage and queries
  - Already used in:
    - LAStools (lassort/lasindex)
    - Oracle blocks (Data preparation in Hilbert R-Tree blocking)
- Space filling curves: Hilbert/Morton
  - Useful for flat model directly
  - X, Y -> Code (position in the curve) No need to store X,Y!
  - Can also be used in cases where point dimension is not per se spatial (x, y, z), but of different nature (t or vario-LoD/imp)
  - Queries need to be re-written to use spatial clustering and be more efficient

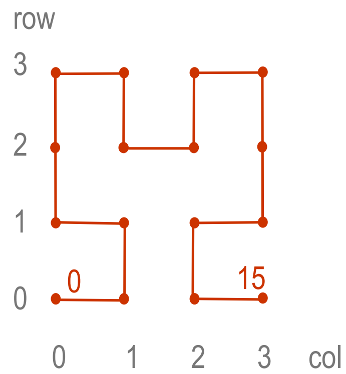
Data organization

Faster flat tables

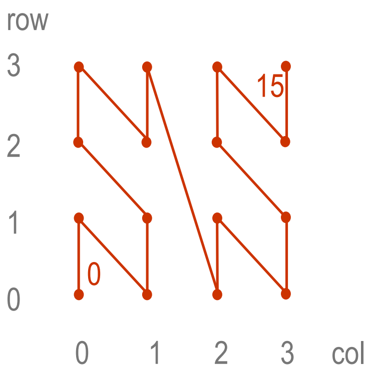
Row (first y, then x)



Hilbert



Peano



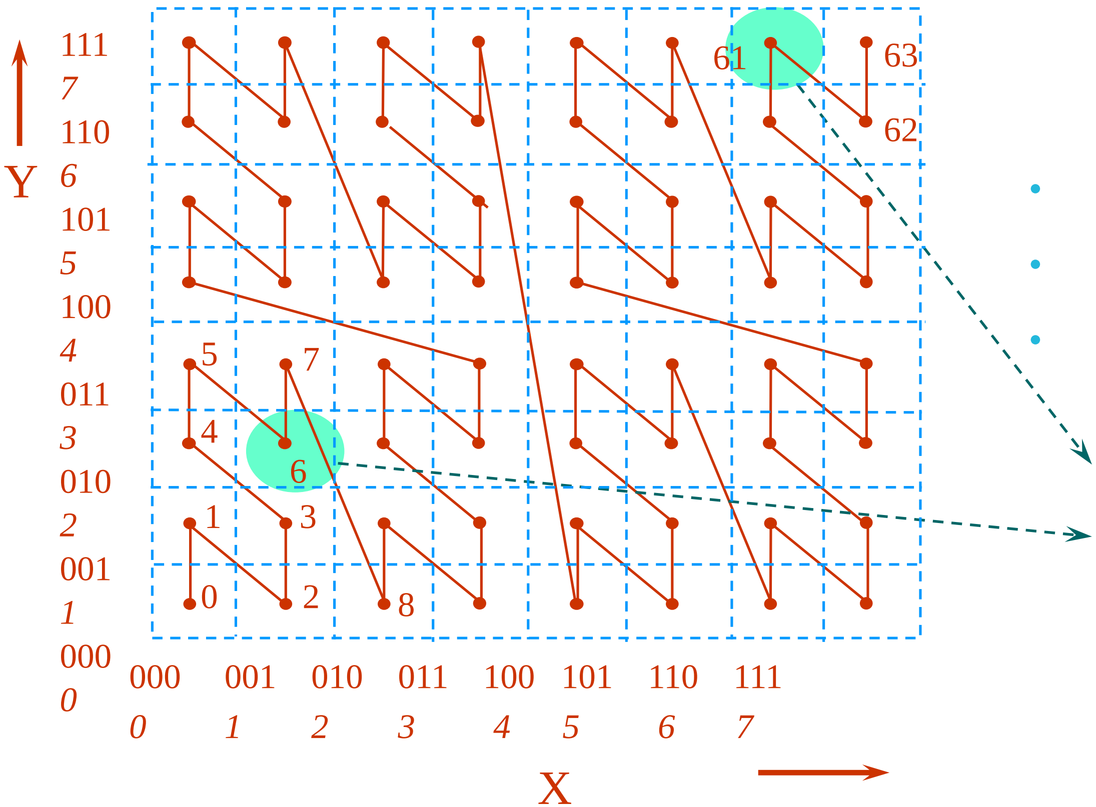
default of flat model

ALTERNATIVE:

- compute Hilbert / Morton codes for all points
- create b-tree index on Hilbert / Morton code (position in the curve)
- cluster flat table on Hilbert / Morton curve
- re-write queries to select ranges of codes

Data organization

Rewriting queries (Morton queries)



- Bitwise interleaving x-y coordinates
- Also works in higher dimensions (nD)
- Two example of Morton code:

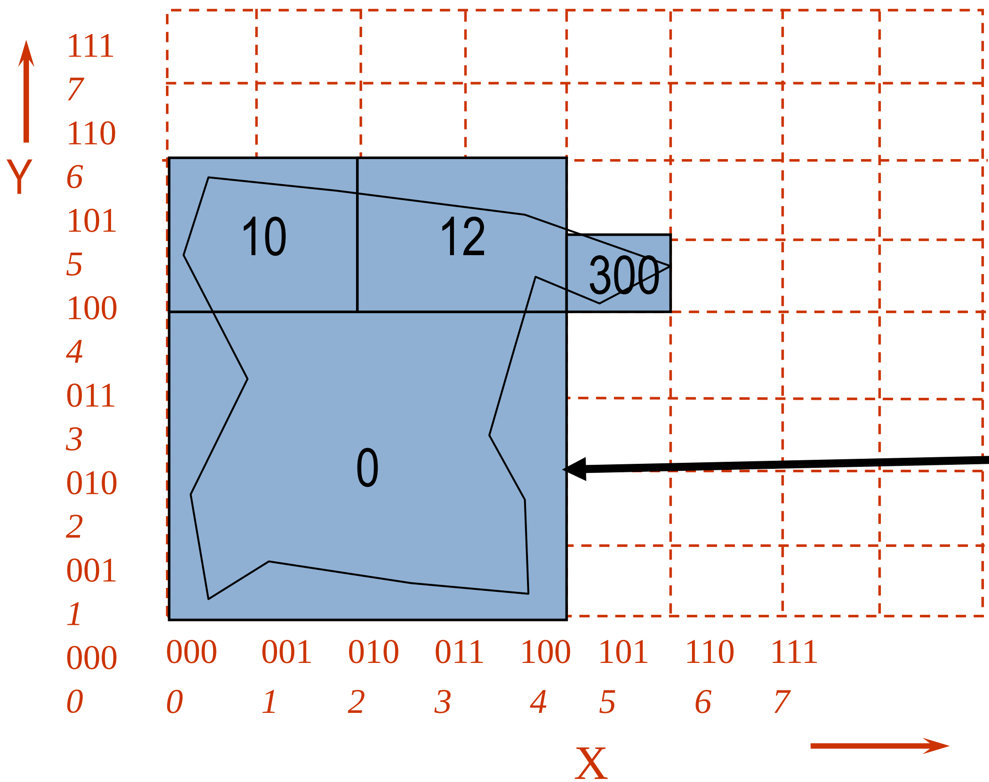
x= 110, y=111 → xy= 111101 (decimal 61)

x= 001, y=010 → xy= 000110 (decimal 6)

Data organization

Rewriting queries (Morton queries)

- Based on concepts of Region Quadtree & Quadcodes
- Works for any type of query geometry
- Also works in 3D (Octree) and higher dimensions



Quadcode 0: Morton range 0-15

Quadcode 10: Morton range 16-19

Quadcode 12: Morton range 24-27

Quadcode 300: Morton range 48-48

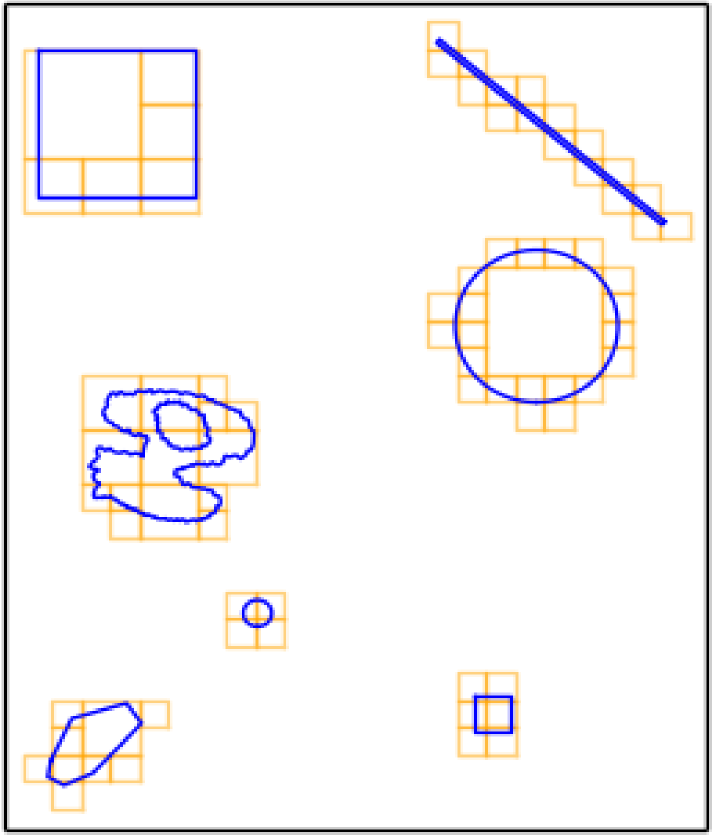
(Morton code gaps resp. 0, 4, 20)

query_geometry, polygon

Note : SW=0, NW=1, SE=2, NE=3

Data organization

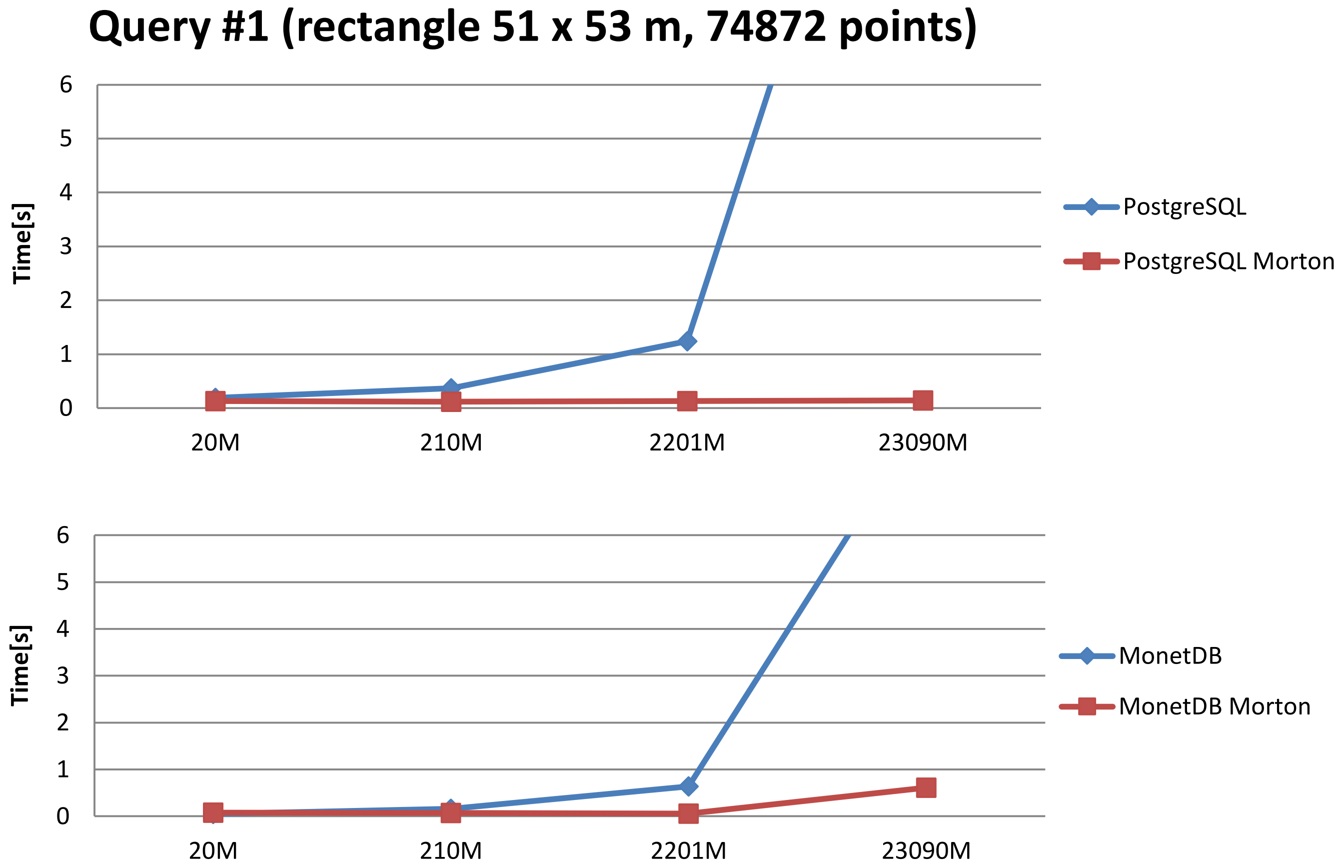
Rewriting queries (Morton queries)



```
CREATE TABLE query_results_1 AS (
    SELECT x,y,z FROM (
        SELECT x,y,z FROM ahn_flat WHERE (
                    (morton2D between 1341720113446912 and 1341720117641215)
               OR (morton2D between 1341720126029824 and 1341720134418431)
               OR (morton2D between 1341720310579200 and 1341720314773503)
               OR (morton2D between 1341720474157056 and 1341720478351359)
               OR (morton2D between 1341720482545664 and 1341720503517183)
               OR (morton2D between 1341720671289344 and 1341720675483647)
               OR (morton2D between 1341720679677952 and 1341720683872255))
    ) a  WHERE (x between 85670.0 and 85721.0)
            and (y between 446416.0 and 446469.0));
```

Data organization

Rewriting queries (Morton queries)



Content overview

1. Introduction
2. Conceptual benchmark
3. Executable benchmark
4. Data organization
5. Conclusion

Summary

- Very innovative and risky project
- No solutions available today (big players active; e.g. Google with street view also collects point clouds, but has not be able to serve these data to users)
- Intermediate results: significant steps forward (explicit requirements, benchmark, improved products,…)
- Direct contact with developers: Oracle, but also MonetDB, PostgreSQL/PostGIS, LAStools,…
- Standardization: discussions started (ISO, OGC)
- Concepts developed for Multi-/vario-scale point clouds (LoD’s, data pyramid)
- parallel query algorithms

Next phases of project

- Full and scaled-up benchmarking
- Web-based viewer (WebGL, LoD-tiles, Fugro prototype)
- Model for operational service (for University users)
- Ambitious project plan, further increased:
  - MonetDB
  - LAStools (and Esri's ZLAS format)
  - Patty project
  - Via Apia project
- More data management platforms (optional):
  - SpatialHadoop
  - MS Azure data intensive cloud (announced last week)/MS SQL server
  - GeolinQ (layered solution with bathymetric/hydrographics roots
- More data?
  - Cyclomedia images / areal photographs
  - Very high density, prediction 35 trillion points for NL
  - More attributes (r,g,b) -> 100 times more data than full AHN2

Future topics (beyond project)

- Possible topics:
  - Different types of hardware/software solutions for point cloud data management (e.g. SpatialHadoop, or LAStools/Esri format tools)
  - Next to multiple-LoD's (data pyramid), explore true vario-scale LoD's
  - Advanced functionality (outside our current scope): surface/ volume reconstruction, temporal difference queries, etc.
  - Higher dimensional point clouds, storing, structuring point clouds as 4D, 5D, 6D, etc points (instead of 3D point with a number of attributes), explore advantages and disadvantages
- Partners (Fugro, RWS or Oracle) most likely interested
- Also interest form others (Cyclomedia, MonetDB)

Thank you!