Università degli Studi di Torino
Facoltà di Fisica
Corso di Laurea in Fisica

Handling of KASCADE Database Queries by means of Indexes

Laureando: Appella Simone
Relatori: Prof. Bertaina Mario Edoardo, Dr. Haungs Andreas
Co - relatori: Schoo Sven, Dr. Kang Donghwa

A.A. 2015/2016

Erasmus Traineeship Location

25/04/2016 – 25/06/2016

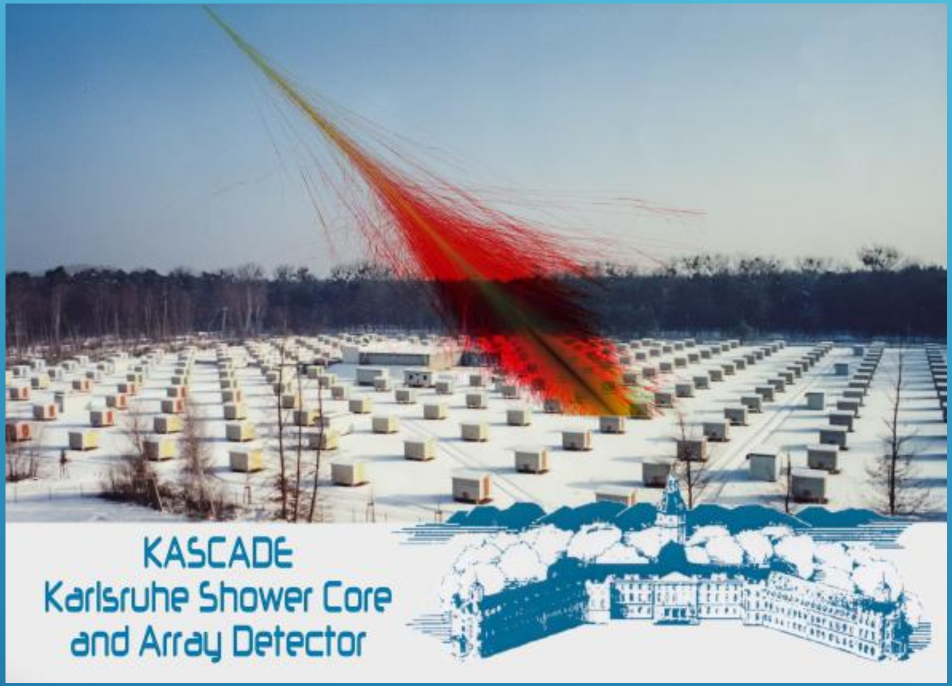
Karlsruher Institut für Technologie

KIT Campus North

Karlsruhe (Baden - Württemberg, Germany)

Work - Team Photo

KASCADE Cosmic Ray Data Centre



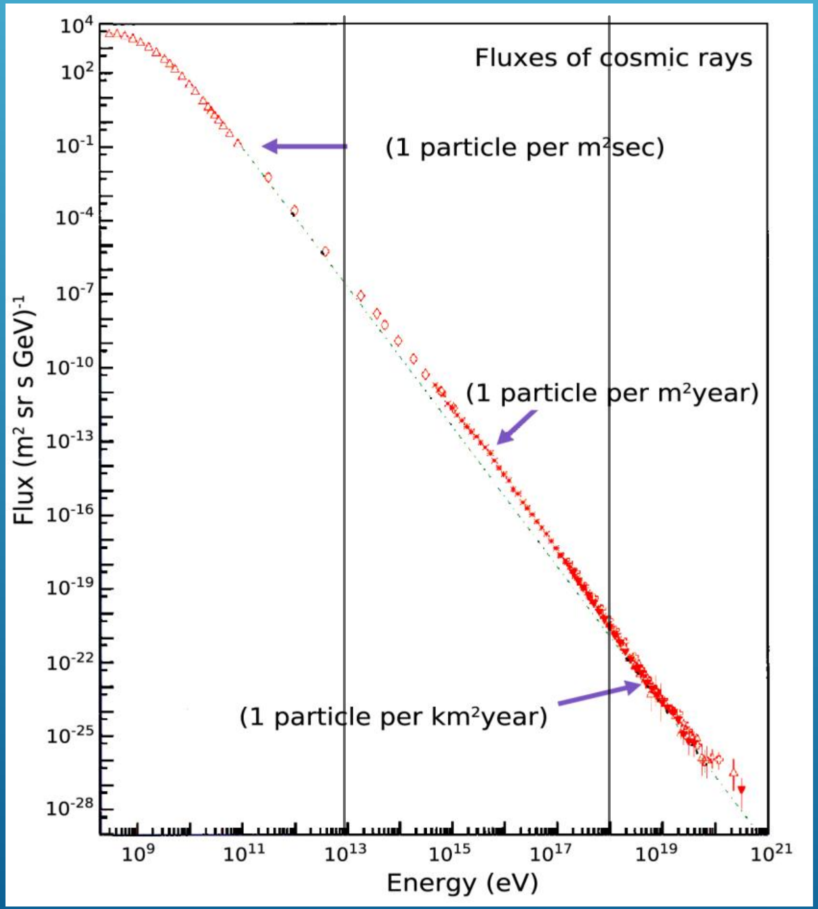
KCDC project aims to create a public cosmic - ray physics data centre holding about **150 million** events from KASCADE experiment, recorded between 1996 and 2012

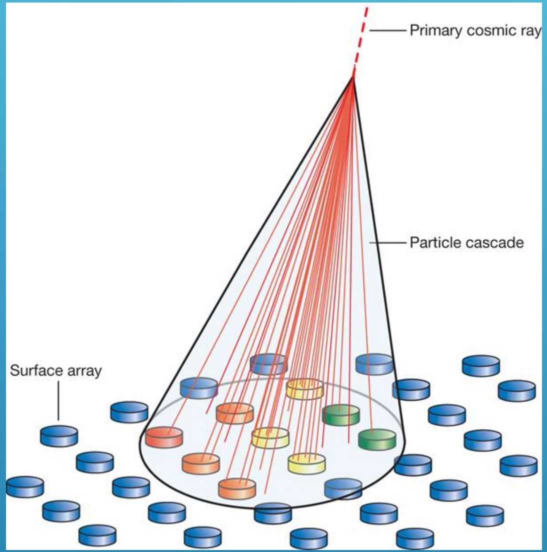
KASCADE experiment acquired data by means of two detectors :

- KASCADE Array
- Hadron Calorimeter

What are Cosmic Rays ?

Cosmic rays are composed of high – energy protons and atomic nuclei coming from different cosmic sources. As they interact with the atmosphere they may produce showers of secondary particles (electrons, muons, hadrons) detectable to ground level.





- $E < 10^{10}$ eV : sources close to Earth
- $10^{12} < E < 10^{17}$ eV : galactic origin
- $E > 10^{18}$ eV : extra - galactic origin

KASCADE Array

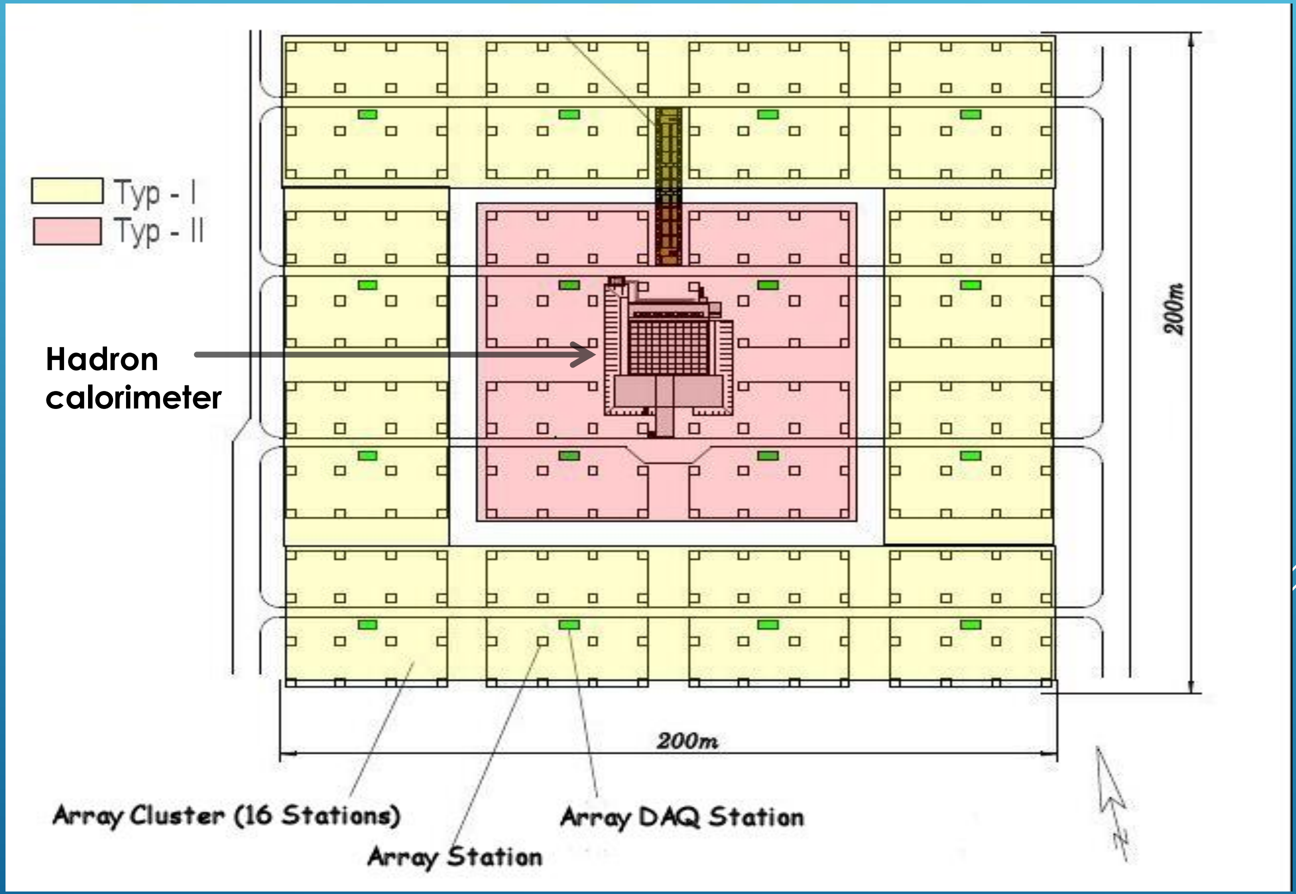
252 detection stations are arranged in 16 clusters. The 12 outer clusters are provided with unshielded liquid scintillators to measure the e/γ components, while muons are detected below 2 absorber sheets. The inner clusters host only liquid scintillators.

$E_{electron}$ threshold : 5 MeV

E_{muon} threshold : 230 MeV

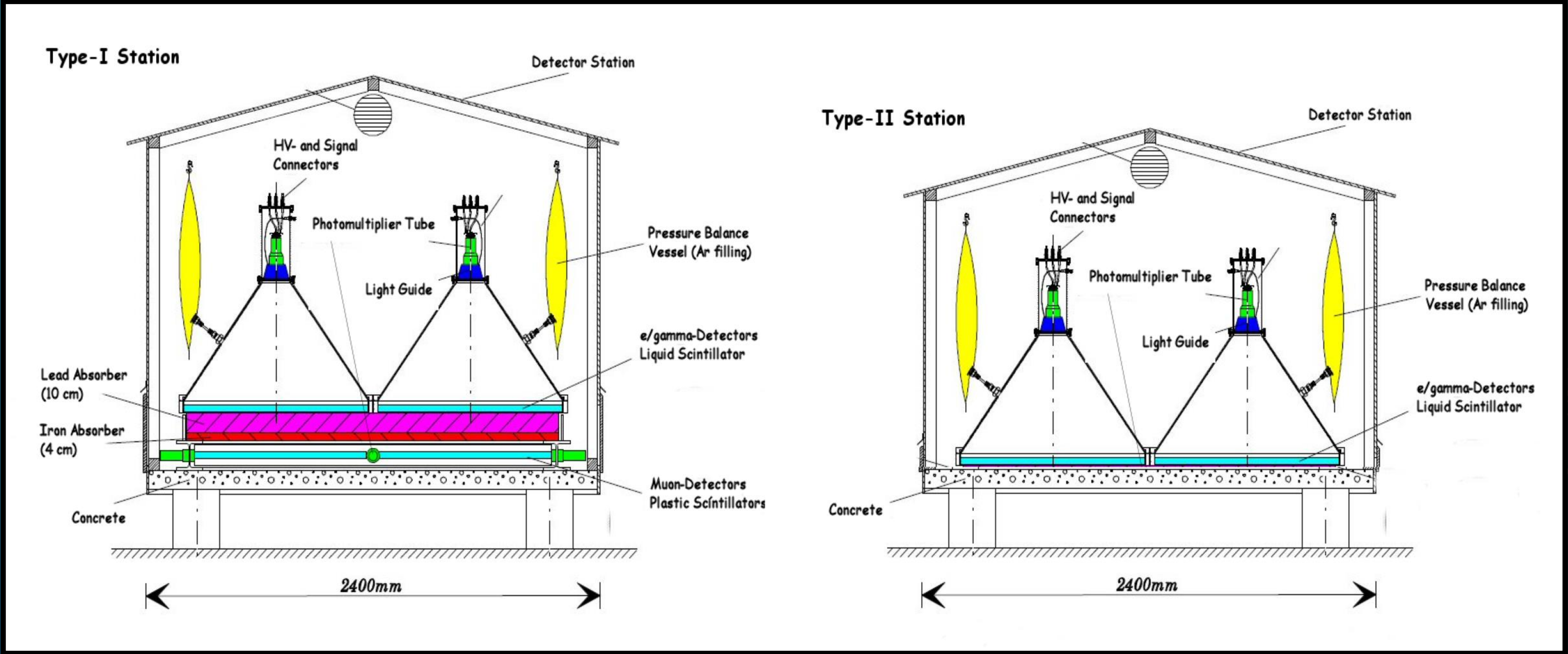
Time resolution : 0.77 ns

Angular resolution : 0,1 °



Area : 200 m^2

Station Set - Up



Schematic view of the Array Detector Stations

KASCADE Parameters

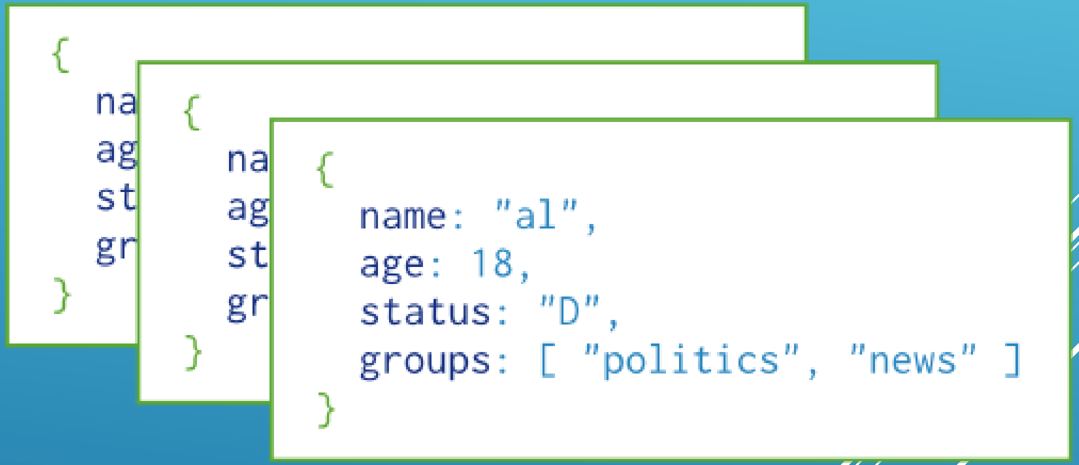
From electron density lateral distribution and arrival time of shower components other parameters, characterizing a single shower event, are reconstructed:

Parameter	Range	Description
Energy (E)	13 – 18 eV [log10]	Primary particle Energy estimated from Ne and Nmu by means of simulations
Number of muons (Nmu)	2 – 7.7 [log10]	Nmu with Energy above 100 MeV
Number of electrons (Ne)	2 – 8.7 [log10]	Ne with Energy above 3 MeV
Age	0.1 – 1.48	Slope parameter of the electron density lateral distribution
Zenith angle (Ze)	0 – 60°	Angle derived from the arrival time distribution
Azimuth angle (Az)	0 – 360°	Angle derived from the arrival time distribution
X, Y shower core position (Xc , Yc)	-91 – 91 m	Reconstructed location of the shower centre
Global time (Gt)	$8.9*10^8$ – $1.07*10^9$ s	A Unix Time counter for seconds elapsed since 1.01.1970.

MongoDB

MongoDB is a database that stores KASCADE events in documents. They are composed of field and value pairs. Each field may include other documents and arrays

```
{
   name: "sue",                          ←  field: value
   age: 26,                              ←  field: value
   status: "A",                          ←  field: value
   groups: [ "news", "sports" ]          ←  field: value
}
```



Collection

Document → Collection → Database

shower event → Meridian → **Meridian_3 / Meridian_3s** (1 million data)

KASCADE Document

⊟ (477) Document (4)	56d56420521811a6298ef7f9...	Document
_id	56d56420521811a6298ef7f9	ObjectId
⊞ general (7)		Document
⊟ array (9)		Document
E	14.701221466064453	Double
Xc	-0.22888532280921936	Double
Yc	10.894941329956055	Double
Ne	4.6545400619506836	Double
Nmu	3.6415719985961914	Double
Age	0.98474907875061035	Double
Ze	15.065551450424932	Double
Az	108.01087666653247	Double
⊞ Stations (252)		Array
⊞ calorimeter (2)		Document
⊞ (478) Document (4)	56d56420521811a6298ef7fa...	Document

Consult the appendix for a full field description

KCDC Data Shop

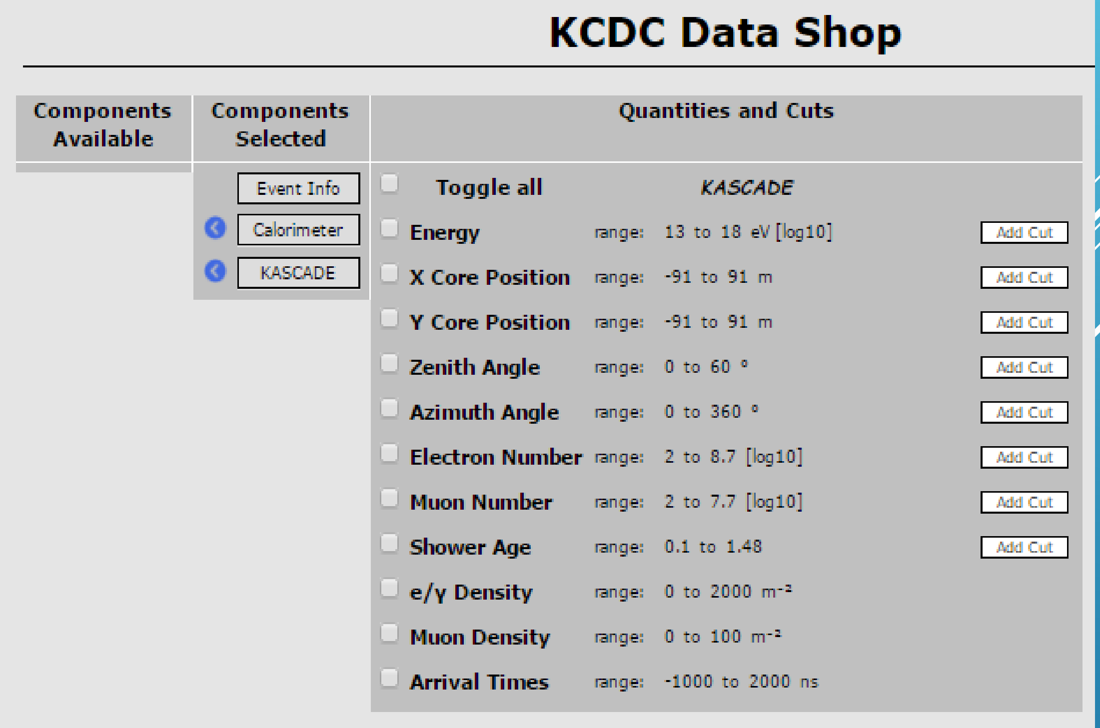
Meridian_3 has stored **150 million** events whose size is about **664 GB.**

Since offering the whole data set takes a great deal of time users are recommended to apply cuts on parameters, which reduce the original file, through KCDC Data Shop

Problem n° 1:

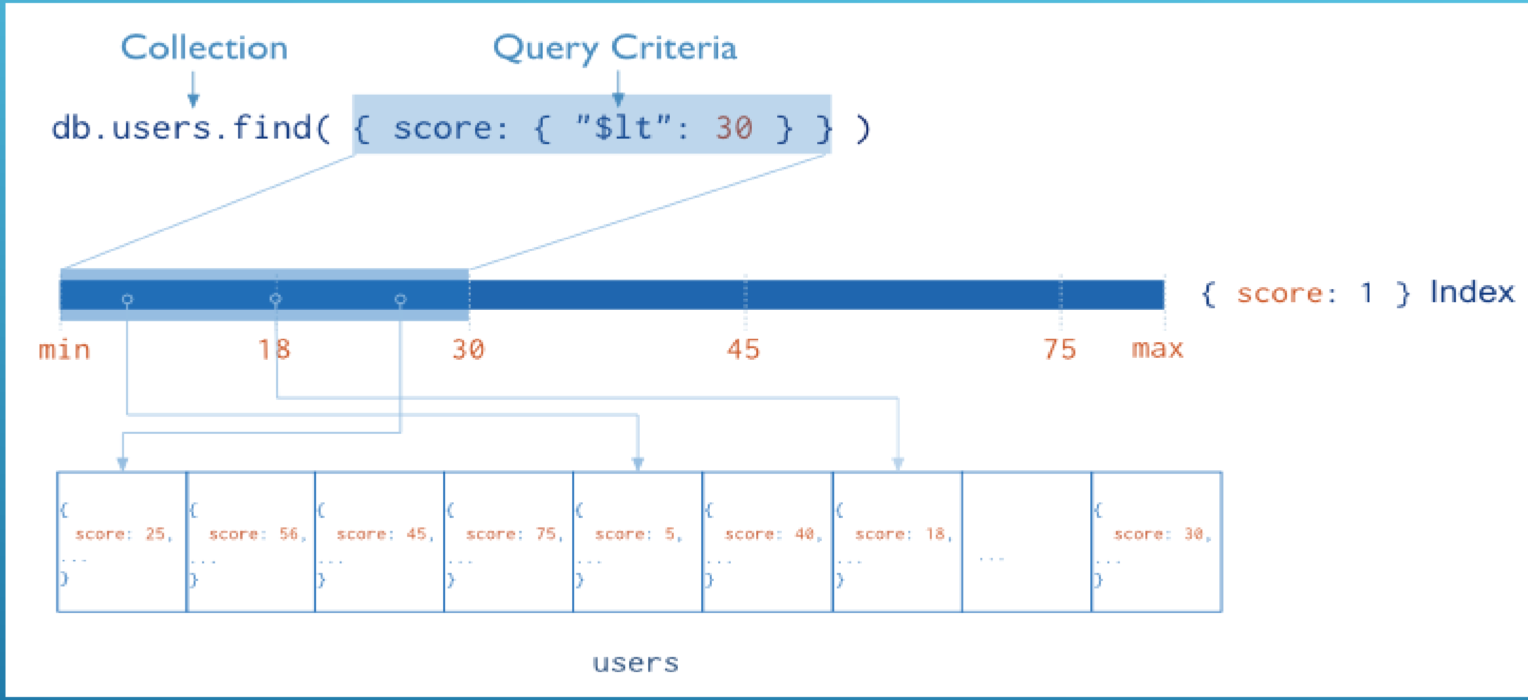
Despite the cuts, MongoDB must perform a collection scan to find docs matching the query criteria. This process takes until **8 hours** !

Single Index



Single Indexes

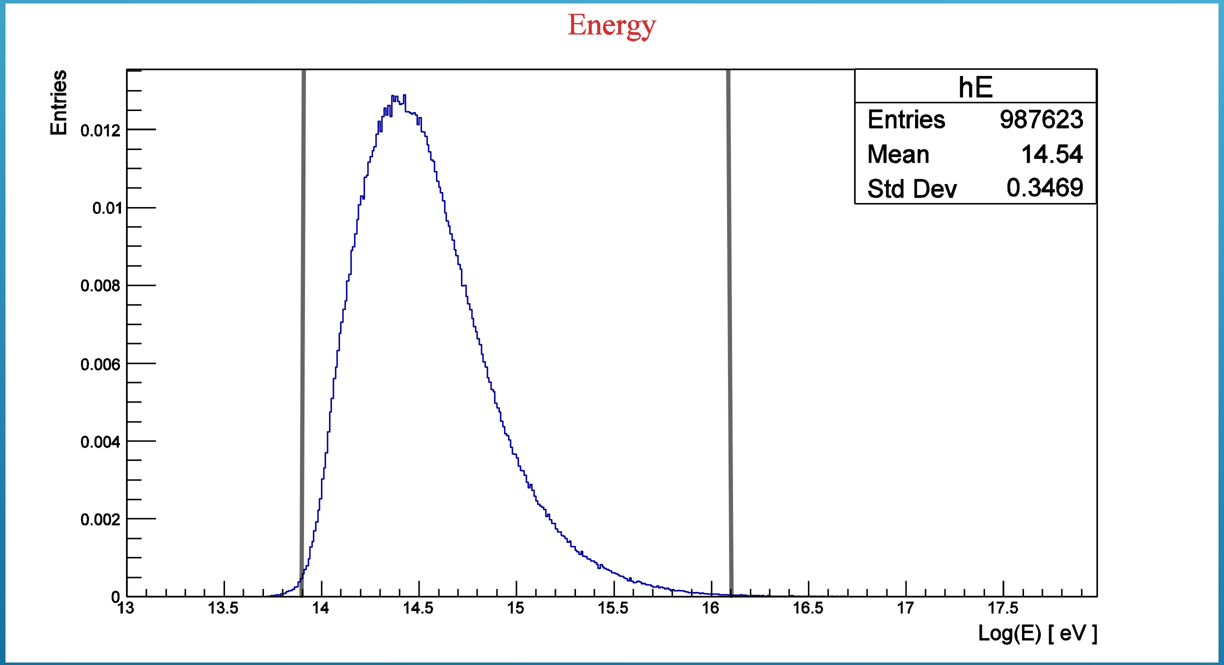
Indexes are data structure, defined at the collection level, that store the values of one field in a certain order. As result the limited scanned documents match automatically the query.



Problem n° 2: **MERIDIAN_3** can not store all possible indexes because they take up considerable RAM space after running a query (~ 1 GB) ⟶ 1 – D histograms

1 – D Histogram Analysis

The most effective indexes must scan as few documents as possible for wide - range cuts. They are selected by analyzing the shape of 1 – D Entry distributions, which derive from a statistically significant sample of 1 million entries stored in **MERIDIAN_3s**.



Number of bins : 500 Bin width : 0.01 eV

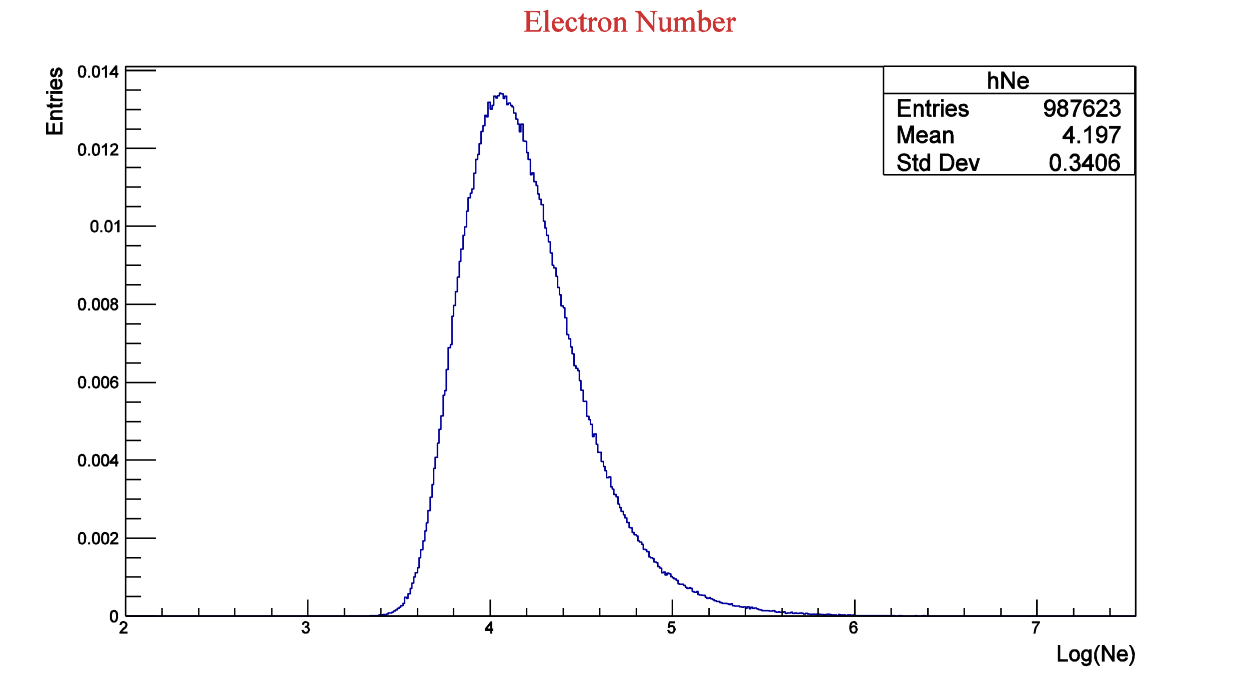
1 - Quality Cut : 13 – 13.9

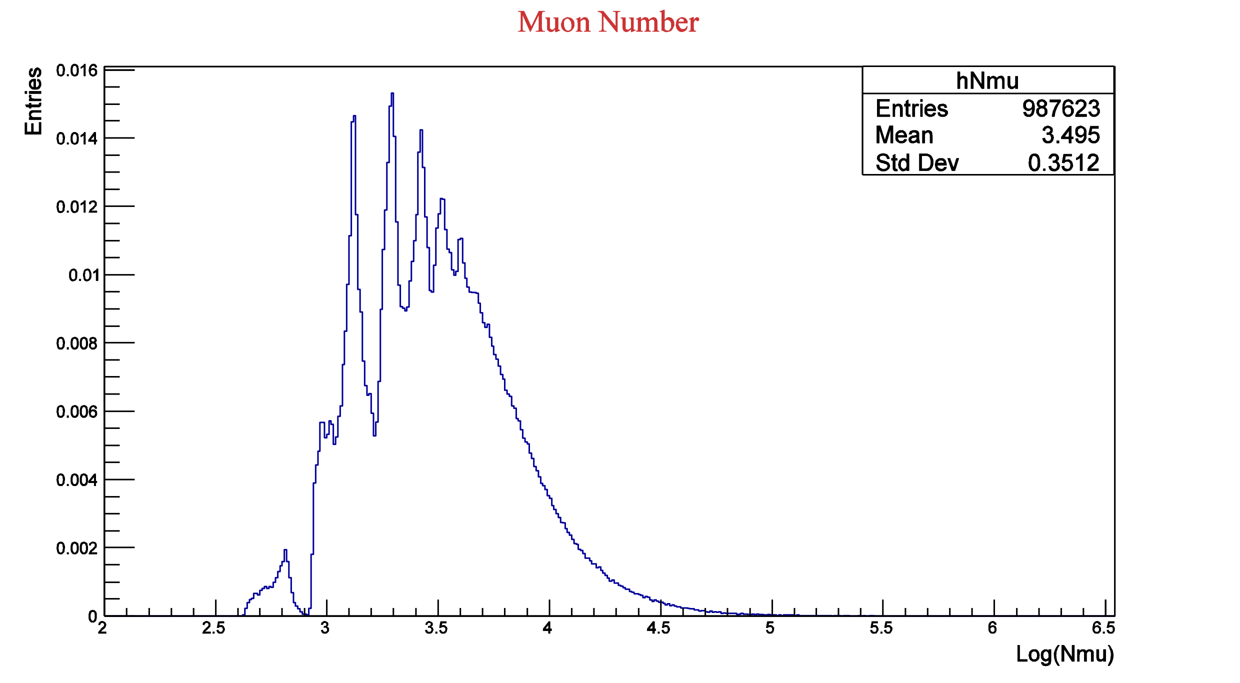
Entries : 1977

2 - Quality Cut : 16.1 – 18

Entries : 1034

1 – D Histograms

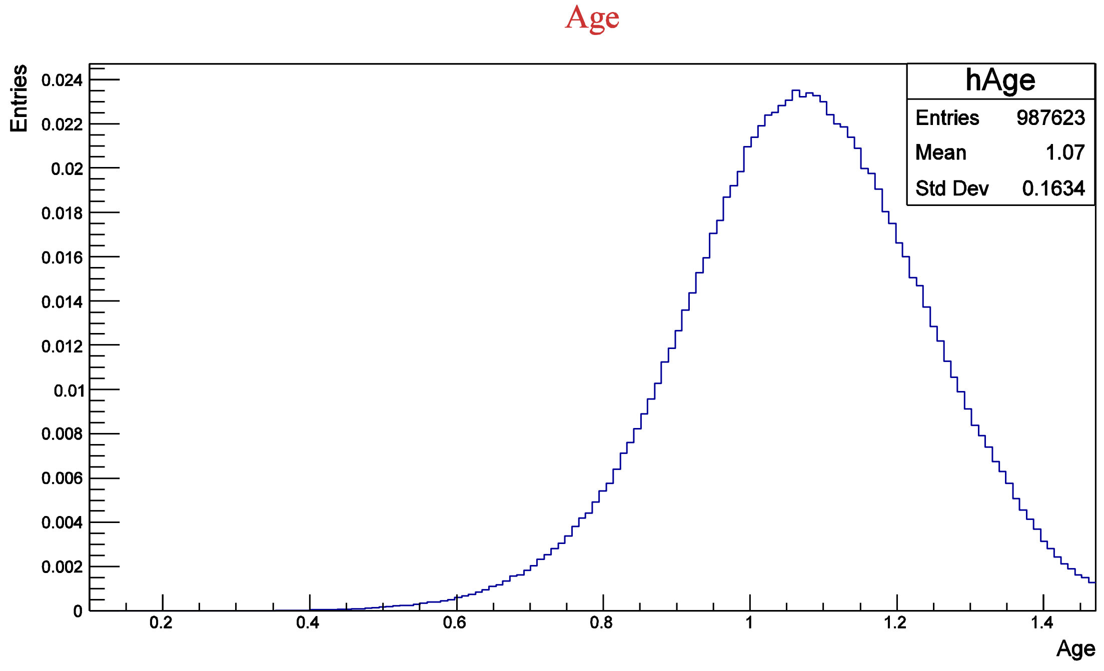


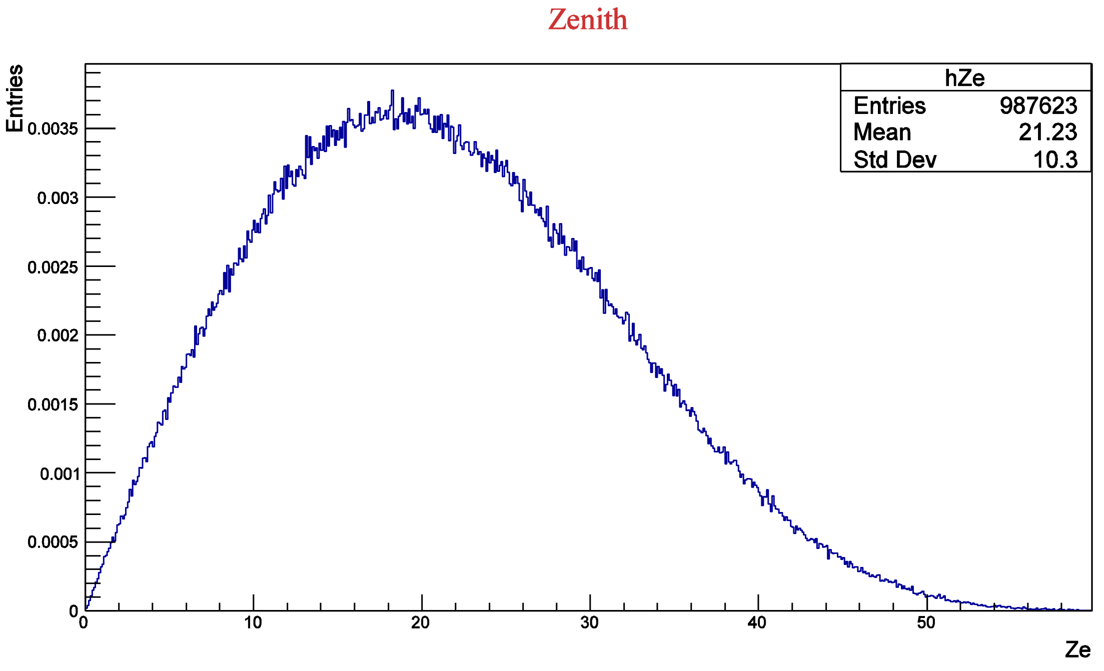


Number of bins : 670 Bin width : 0.01

Number of bins : 570 Bin width : 0.01

1 – D Histograms

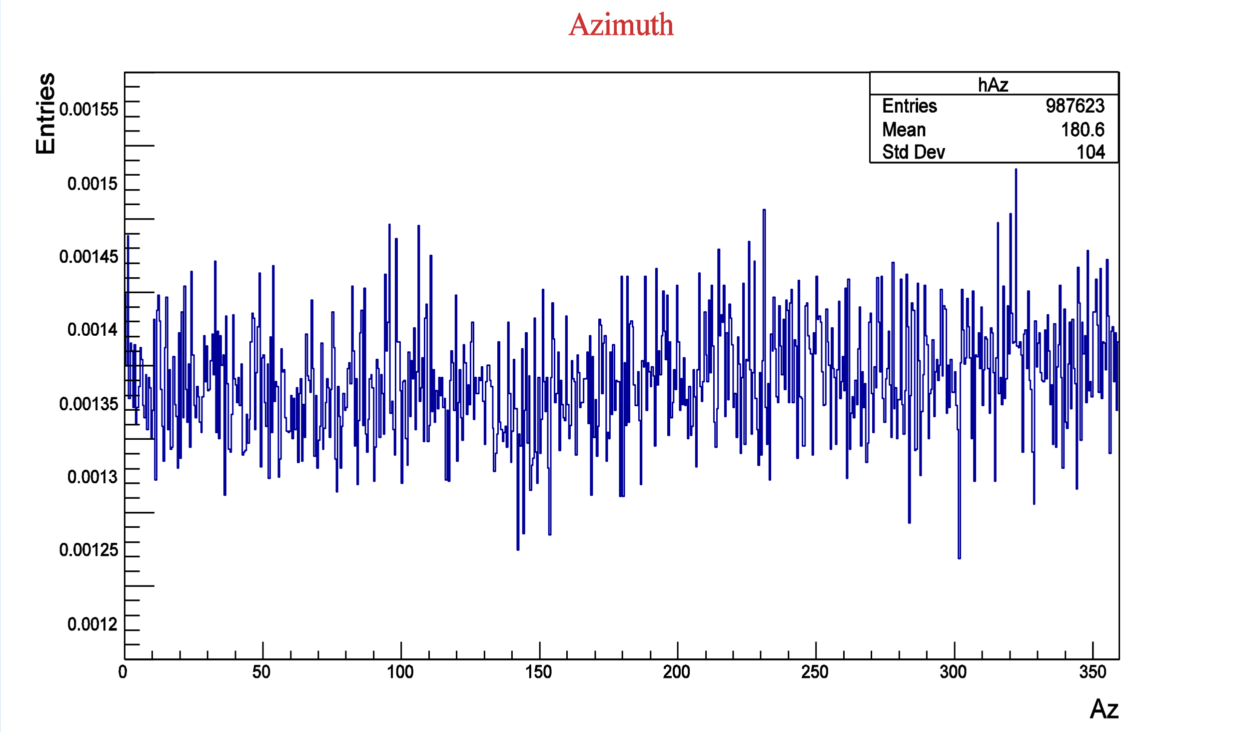


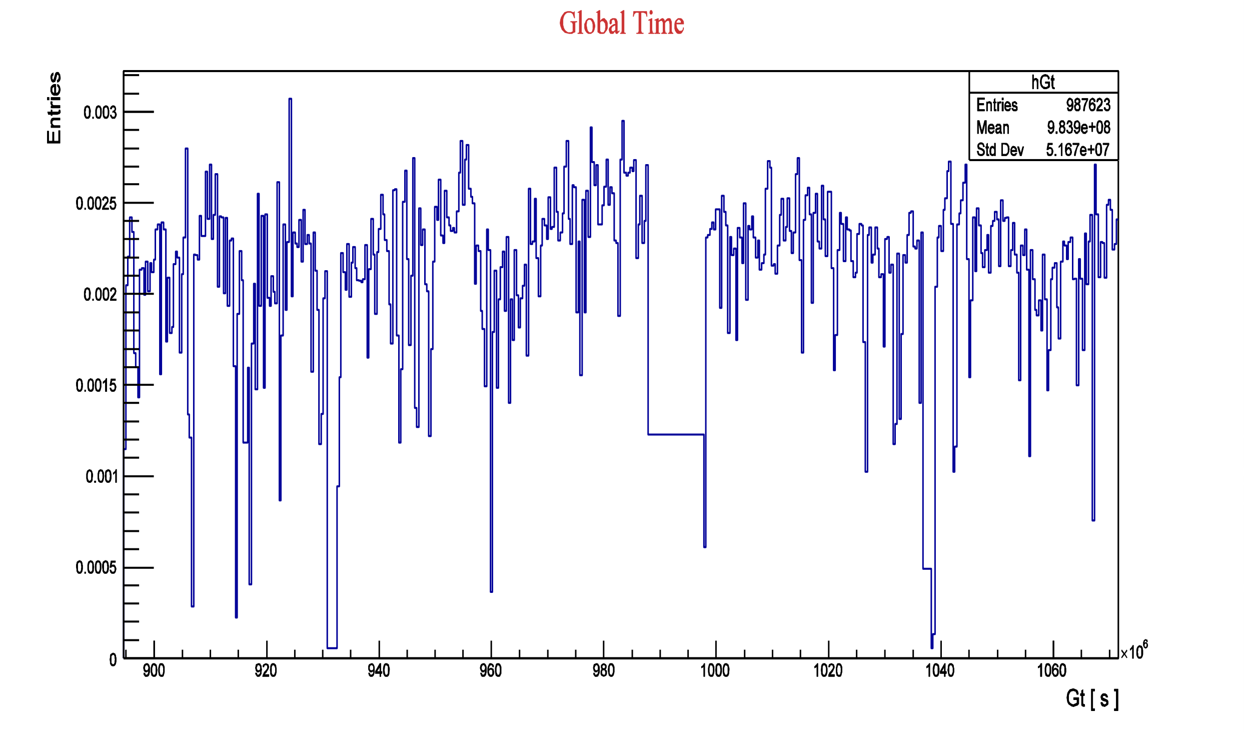


Number of bins : 147   Bin width : 0.01

Number of bins : 600   Bin width : 0.1 °

1 – D Histograms

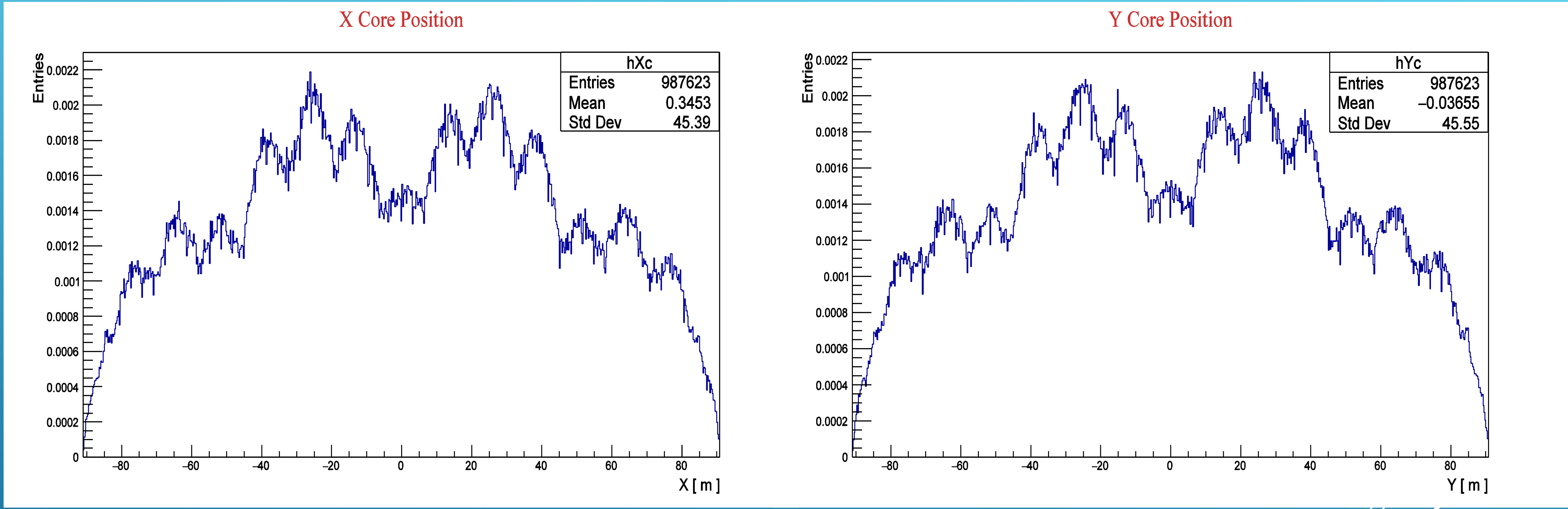




Number of bins : 720   Bin width : 0.5 °

Number of bins : 500   Bin width : 0.35 s

1 – D Histograms



Number of bins : 728   Bin width : 0.25 m

Summary Table

Comparison of smallest intervals containing a minimum of 1000 entries

Parameter	N bins	N tot bins	LowEdge	UpEdge	% Range	Entries
E	90	500	13,00	13,90	18,00	1977
	190	500	16,10	18,00	38,00	1034
Nmu	70	570	2,00	2,70	12,28	2928
	280	570	4,90	7,70	49,12	1176
Ne	160	670	2,00	3,60	23,88	6941
	290	670	5,80	8,70	43,28	1015
Age	50	147	0,10	0,60	34,01	4322
Ze	70	600	53,00	60,00	11,67	1160
Az	2	720	0,00	1,00	0,28	1429
	2	720	359,00	360,00	0,28	1397
Xc	8	500	-91,00	-89,00	1,60	1262
	8	500	89,00	91,00	1,60	1571
Yc	8	728	-91,00	-89,00	1,10	1223
	8	728	89,00	91,00	1,10	1476
Gt	3	500	9,30E+08	9,31E+08	0,57	4039
	6	500	1,04E+09	1,04E+09	1,14	6813

Results on single indexes

- Az , Xc , Yc , Gt

The entries are smeared out over the full range, so a short cut must be applied to decrease them

Deprecated indexes

- E , Nmu , Ne , Age , Ze

The entries are gathered in a bell – shaped distribution, so wide - range cuts close to the borders are restrictive

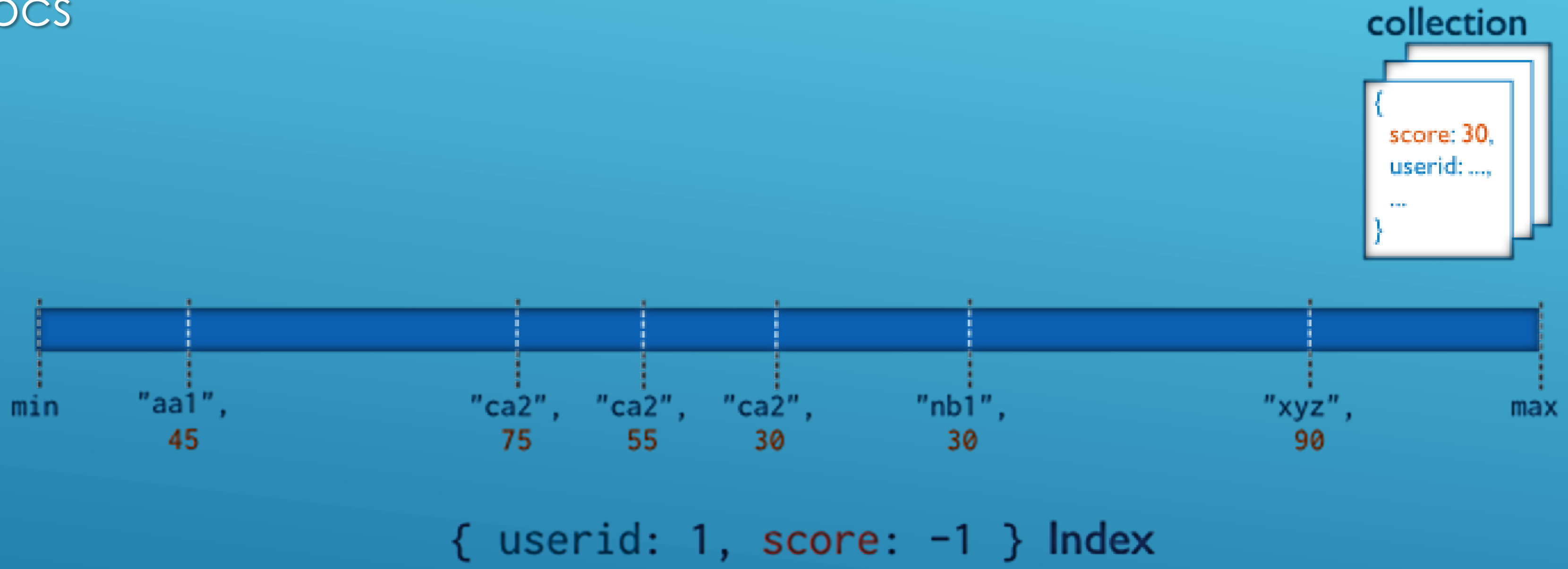
Effective indexes

Problem n° 3:

How to handle queries with two parameter cuts ? → Compound Indexes

Compound Indexes

MongoDB supports compound indexes, that hold references to multiple fields within a collection. The first field has precedence in the order and affects the total number of scanned docs



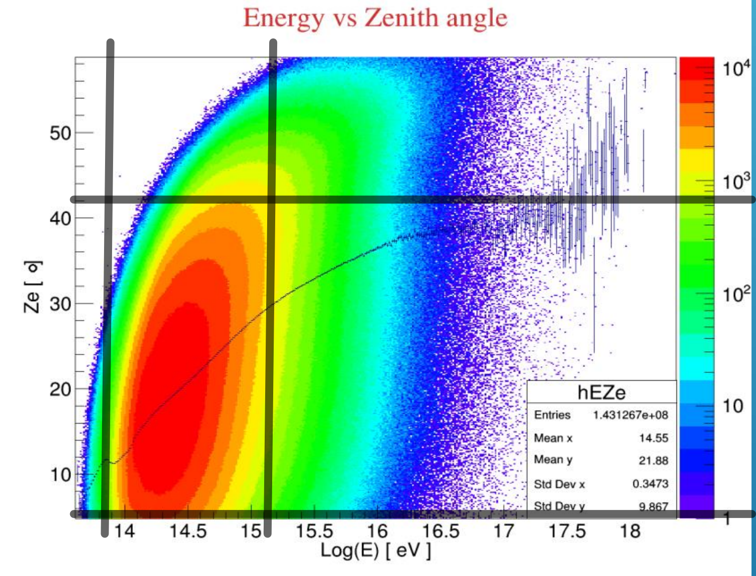
db.users.find({ userid : { $lt : ca2 } , score : { $gt : 40 } })

{ userid : 1 , score : -1 } → 1 Docs scanned → Best compound index

{ score : -1 , userid : 1 } → 4 Docs scanned

2 – D Histogram Analysis

For each parameter couple the order has been selected by analyzing 2 – D distributions. The first field, which is the most restrictive, must fully cover red area (scan most of entries) with less probablity than the other field.

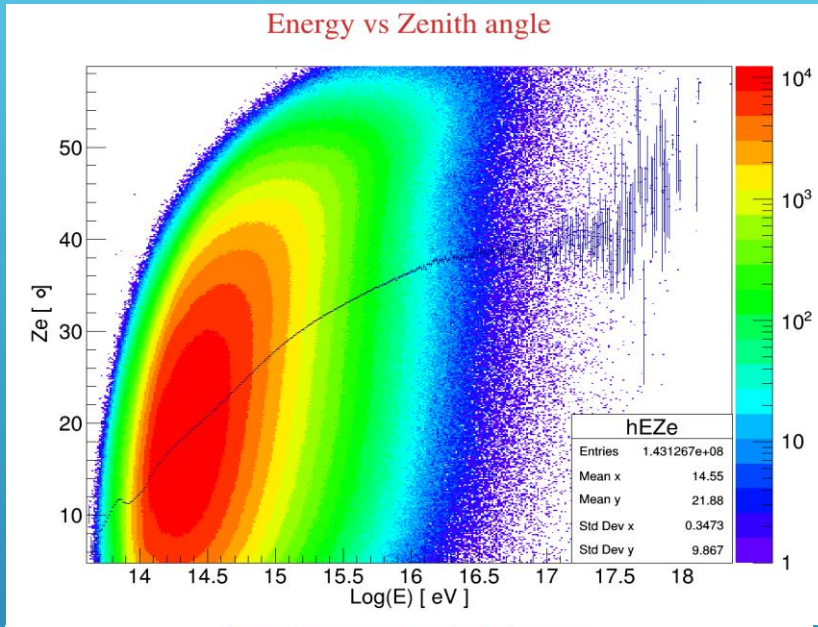


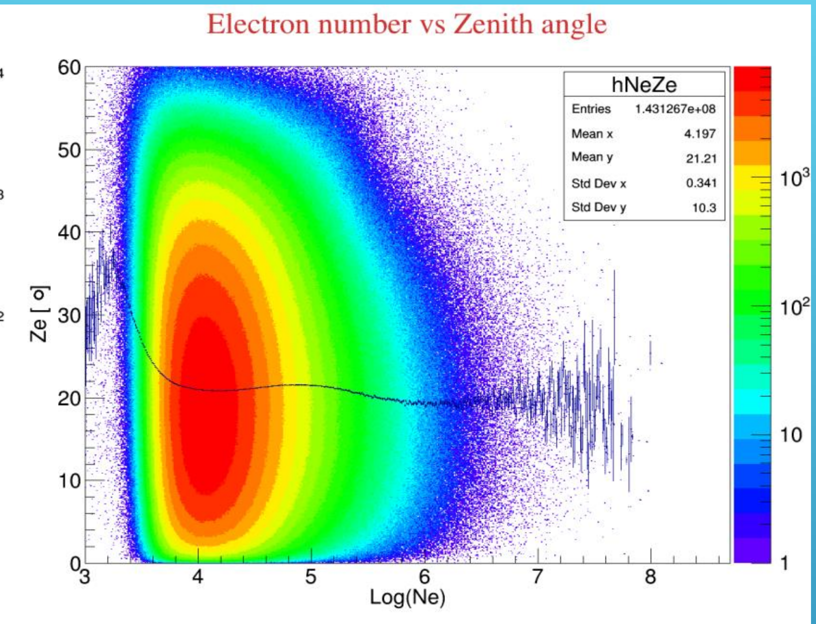
Parameter	Range	% tot Range	Order
Ze	0 - 41	66.7	1
E	13.8 – 15.2	28	2

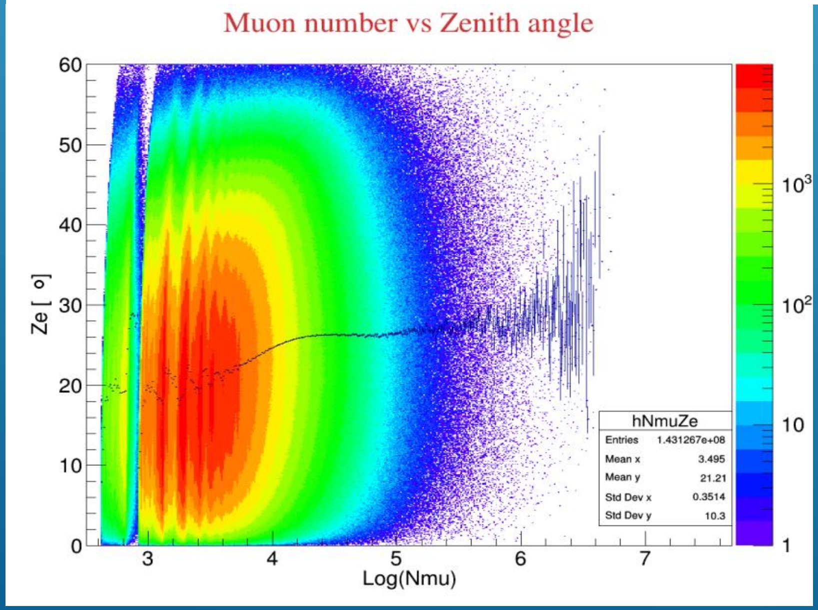
E range might be included in lots of wider cuts !

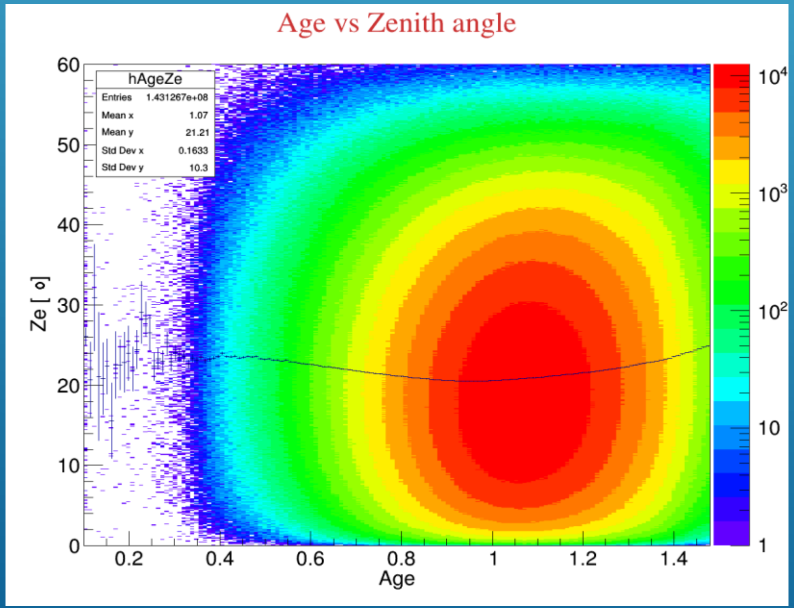
The following Histograms take into account the full data set stored in **MERIDIAN_3** in order to figure out the inter - relation even for high – energy events

2 – D Histograms





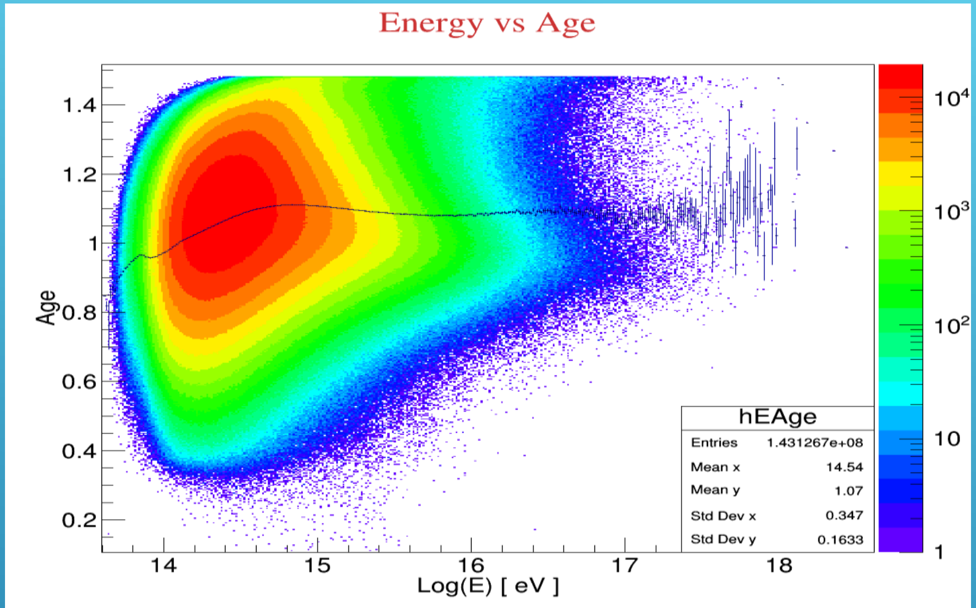




Index : Ze – E, Nmu, Ne

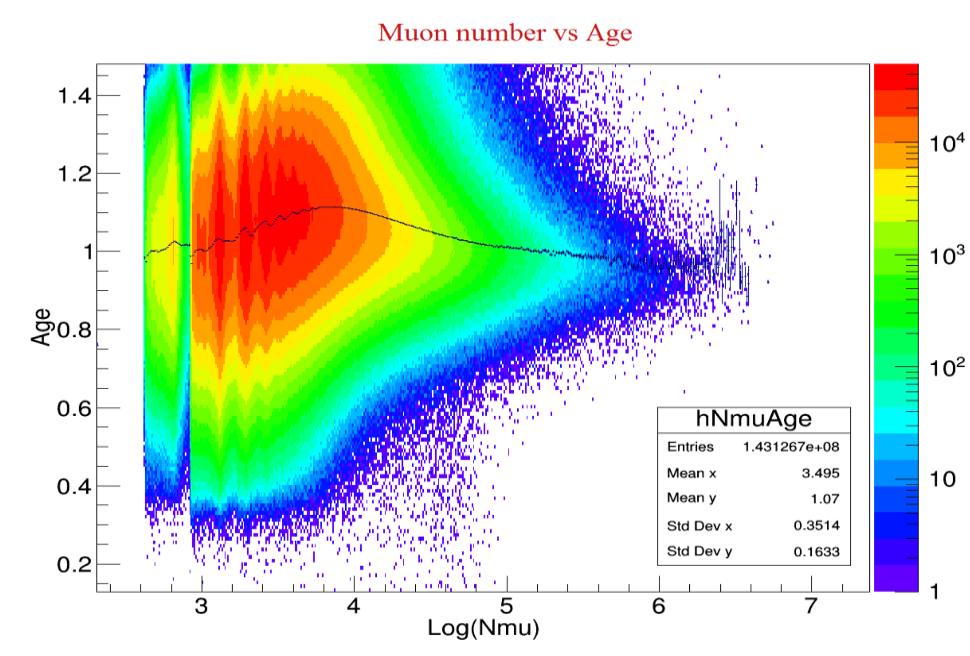
- Entries are vertically stretched
- Index on Age - Ze is not effective because red area is widely spread out

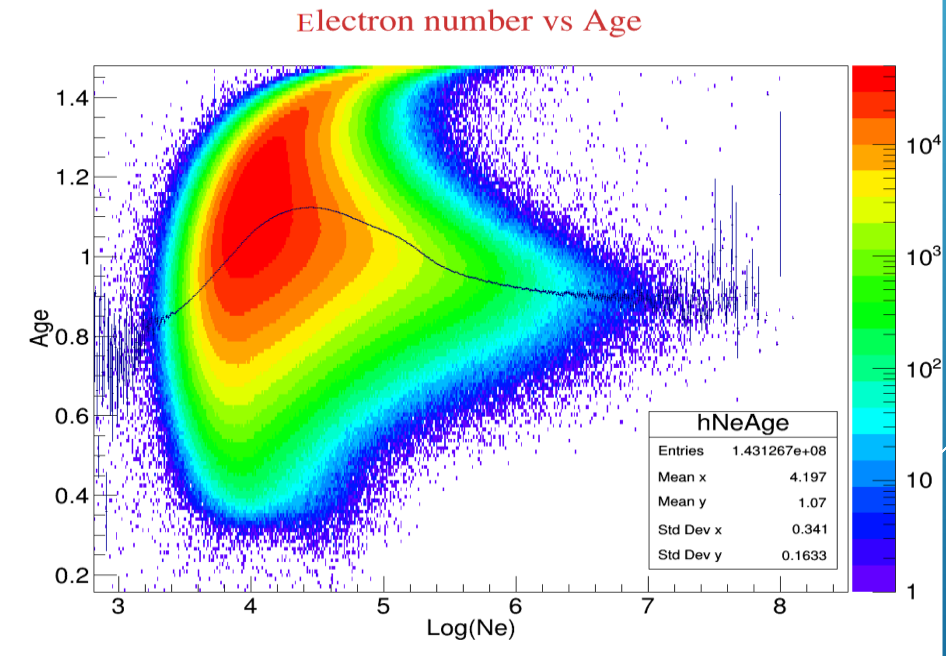
2 – D Histograms



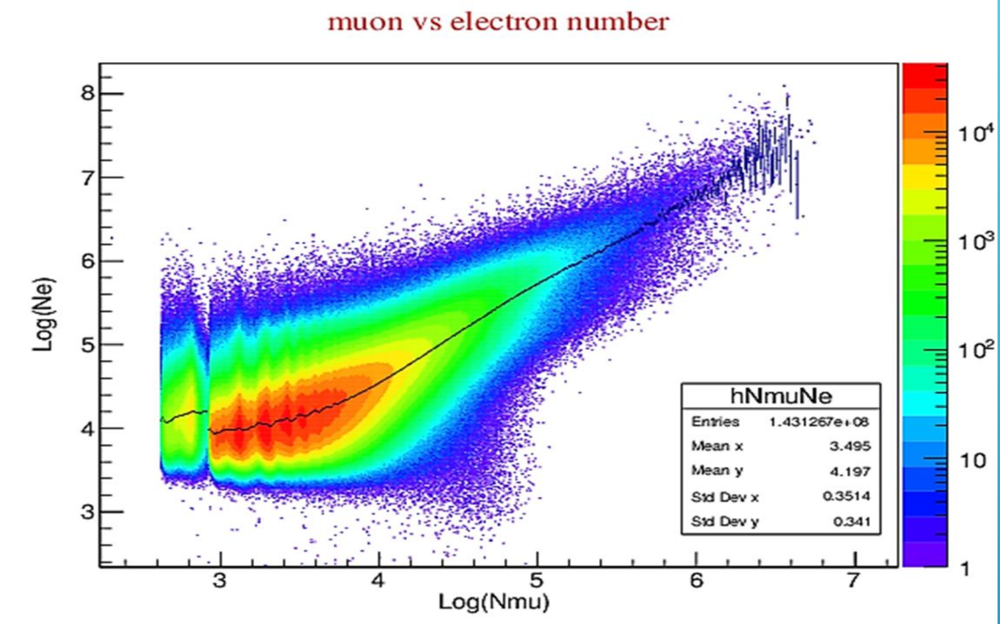
Index : Age – E, Nmu, Ne

- Most of data are gathered in a vertical - shape region
- Horizontal stretching is not relevant since hosts few records
- High – energy events collected for Age values : 0.8 - 1





2 – D Histograms



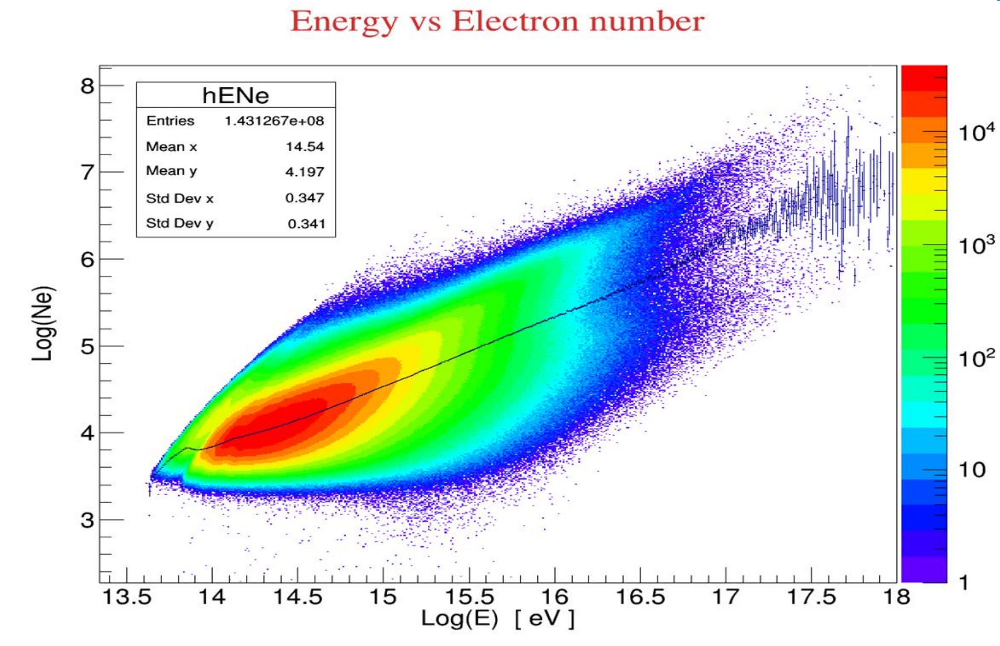
- Strong correlation for high – energy events

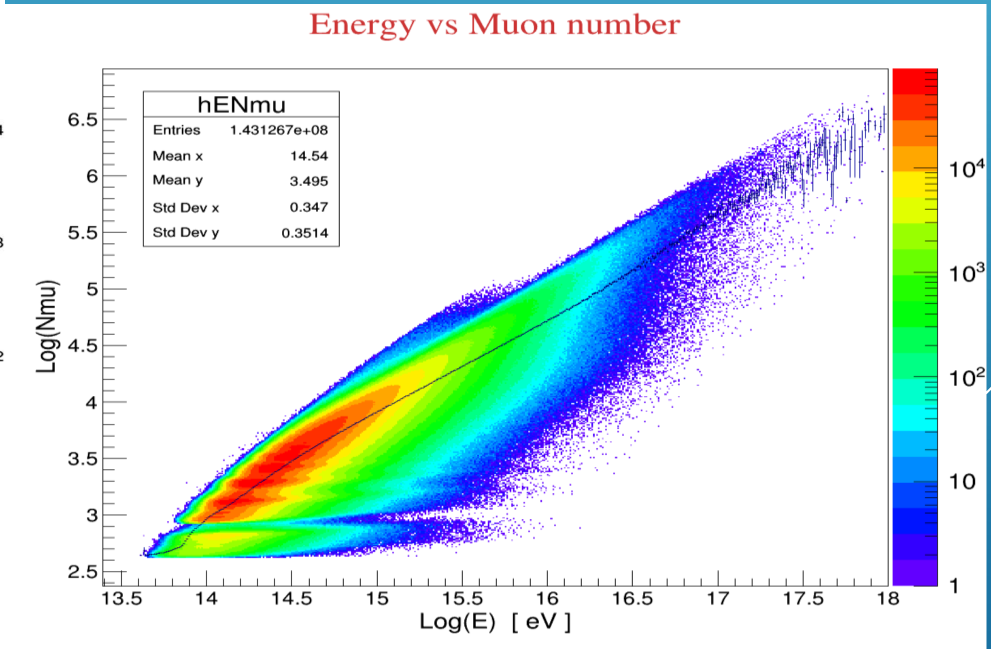
Index : E – Nmu / Nmu - E

- Red area is located along the bisector, index order is not relevant

Index : E – Ne , Nmu - Ne

- Red area is oriented to X axis, cuts on E and Nmu are more restrictive





Query Optimizer vs C ++ Script

Once single and compound indexes are created, the MongoDB algorithm Query Optimizer runs concurrently all of them to flag the first index that collects 100 docs as the most efficient.

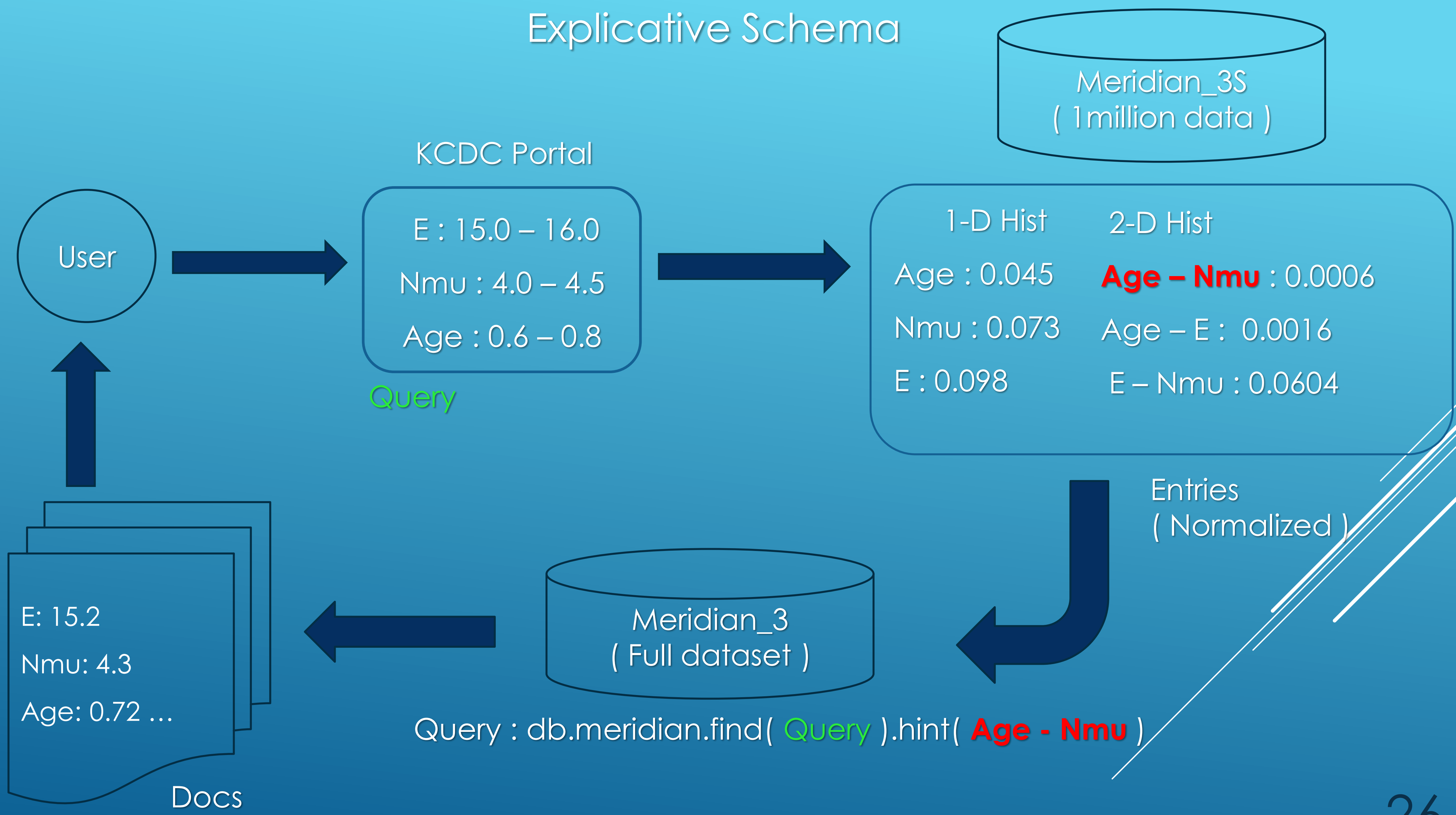
Problem n° 4:

- Number of docs is statistically insignificant compared to the full dataset (150 million), so another existing index might fit better the query statesment
- Concurrent index scan takes time

↓

Query Optimizer has been replaced with a C ++ script, whose workflow is described in the next schema, to extremely reduce the execution time.

Explicative Schema



Execution time comparison

N Query	Parameter	Range	Docs Returned	Script: Index used	Script: Execution Time (ms)	Script: Docs examined	Query Optimizer: Index used	Query Optimizer: Execution Time (ms)	Query Optimizer: Docs examined
1	E	16.5 - 17.0	16	E_Ne	291 ± 3	16	E_Ne	5870 ± 80	16
	Ne	4.0 - 5.0							
2	E	16.0 - 17.0	588	Age_E	10100 ± 100	588	E_Ne	26360 ± 80	1592
	Age	0.8 - 1.0							
3	E	15.5 - 16.0	6212	E_Nmu	77220 ± 360	6212	E_Ne	141240 ± 370	12805
	Nmu	4.0 - 4.5							
4	E	16.0 - 18.0	372	Nmu_Ne	5990 ± 80	402	Nmu_Ne	11350 ± 140	402
	Nmu	5.0 - 7.0							
	Ne	5.0 - 6.0							
5	E	15.3 - 16.0	279	Age_E	8080 ± 230	491	E_Nmu	373430 ± 660	20794
	Nmu	4.0 - 4.5							
	Age	0.6 - 0.8							
6	E	15.5 - 16.0	321	E_Nmu	10090 ± 70	669	E_Nmu	103440 ± 80	669
	Nmu	3.0 - 4.0							
	Ne	4.0 - 5.0							
	Az	120- 360							

Execution times referred to data set sample in Meridian_3s / RAM refreshed after each query

Conclusion

My work is composed of these steps :

- 1 - D entry distribution analysis → Creation of Single Indexes on KASCADE parameters
- 2 – D entry distribution analysis → Creation of Compound indexes
- C++ Script:
  - Get Parameter name, N cuts, cut Edges
  - Count normalized Entries by means of Histograms derived from **Meridian_3s**
  - Order Parameters according to the number of Entries
  - Select the first parameter that is indexed in **Meridian_3**
  - Run a single query on **Meridian_3** without concurrent scan
  - Retrieve docs and send them to the user
  - Save a hu**MONGO**us amount of time !

Thanks for your attention

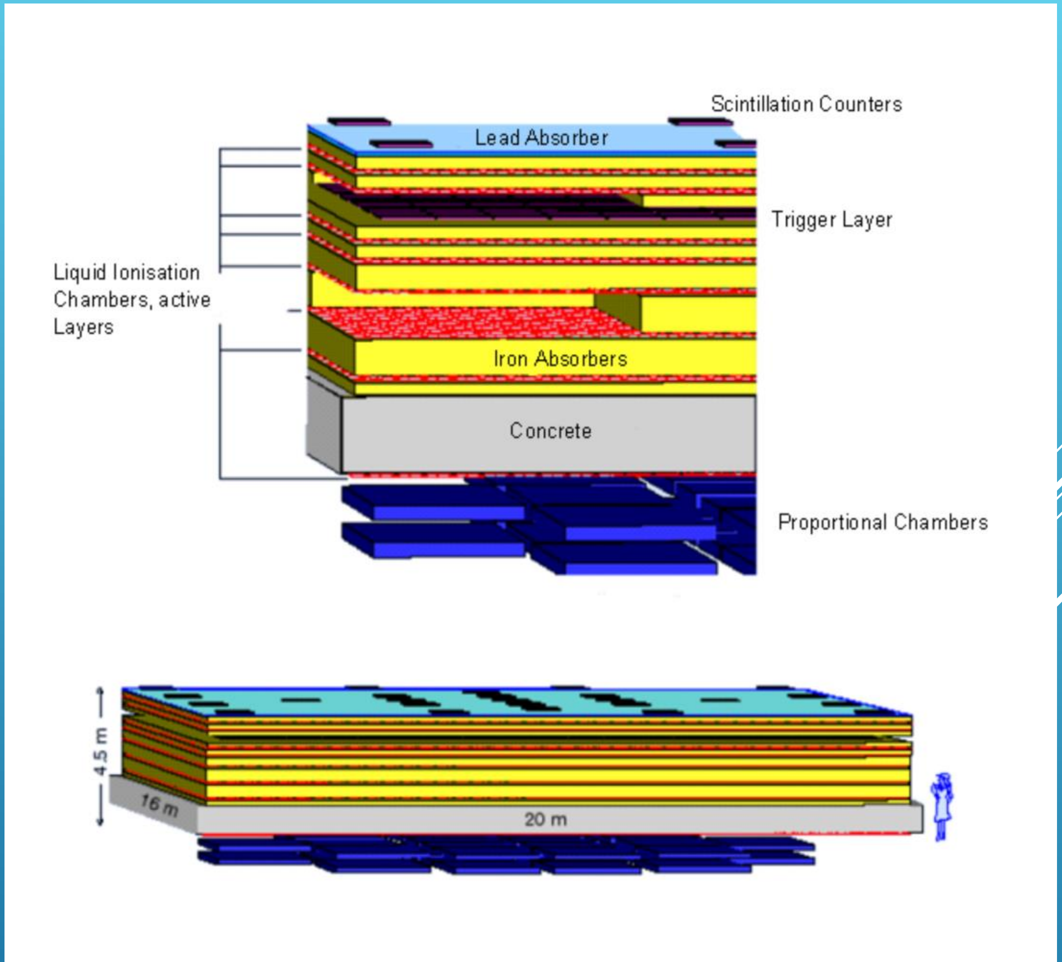
Appendix

Hadron Calorimeter

It is composed of absorber layers and liquid ionisation chambers to reveal hadronic air shower components. The parameters reconstructed are:

- Number of Hadrons (N_{had})
- Hadron Energy (E_{had})

Proportional Chambers below Concrete layer measure position and angle for high energetic muons (E > 2.4 GeV)



Area: 16 × 20 m²

E_{had} resolution: 30 % (100 GeV) – 15 % (25 TeV)

Spatial resolution: 11 cm

Angular resolution: 5 °

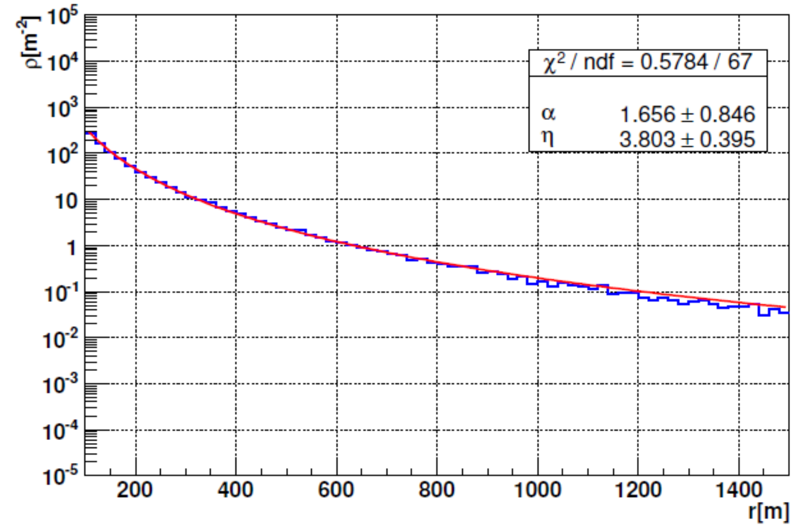
Age

The name Lateral Shower Age Parameter (LSAP) expresses the relation between the shape of the electron density lateral distribution and air shower evolutionary stage.

$$s(r) = \frac{2 - \alpha + (6.5 - \eta)r}{(1 + 2r)}$$

$$\rho(R) = C(\alpha,\eta)\frac{N}{R_0^2}\left(\frac{R}{R_0}\right)^{-\alpha}\left(1+\frac{R}{R_0}\right)^{-(\eta-\alpha)}$$

- α,η : obtained by fitting lateral density distribution ρ(r)
- r : reference distance from shower core estimated by Monte Carlo simulations



Age > 1 → Old shower (origin in upper atmosphere)

Age < 1 → Young shower (origin in lower atmosphere)

- N: total number of shower secondaries
- C: normalization costant
- R0: Moliere unit

Relational Database (RDBMS)

Relational database represents data into one or more tables made up of columns (keys) and rows (records). All records must have the same keys, and each key represents a field to fill in.

Students Table

Student	ID*
John Smith	084
Jane Bloggs	100
John Smith	182
Mark Antony	219

Activities Table

ID*	Activity1	Cost1	Activity2	Cost2
084	Tennis	$36	Swimming	$17
100	Squash	$40	Swimming	$17
182	Tennis	$36		
219	Swimming	$15	Golf	$47

Relationships combine data tables that share a common key.

Why use Mongo DB ?

- Schema – free : Docs can hold different fields in the same collection. No table representation as RDBS

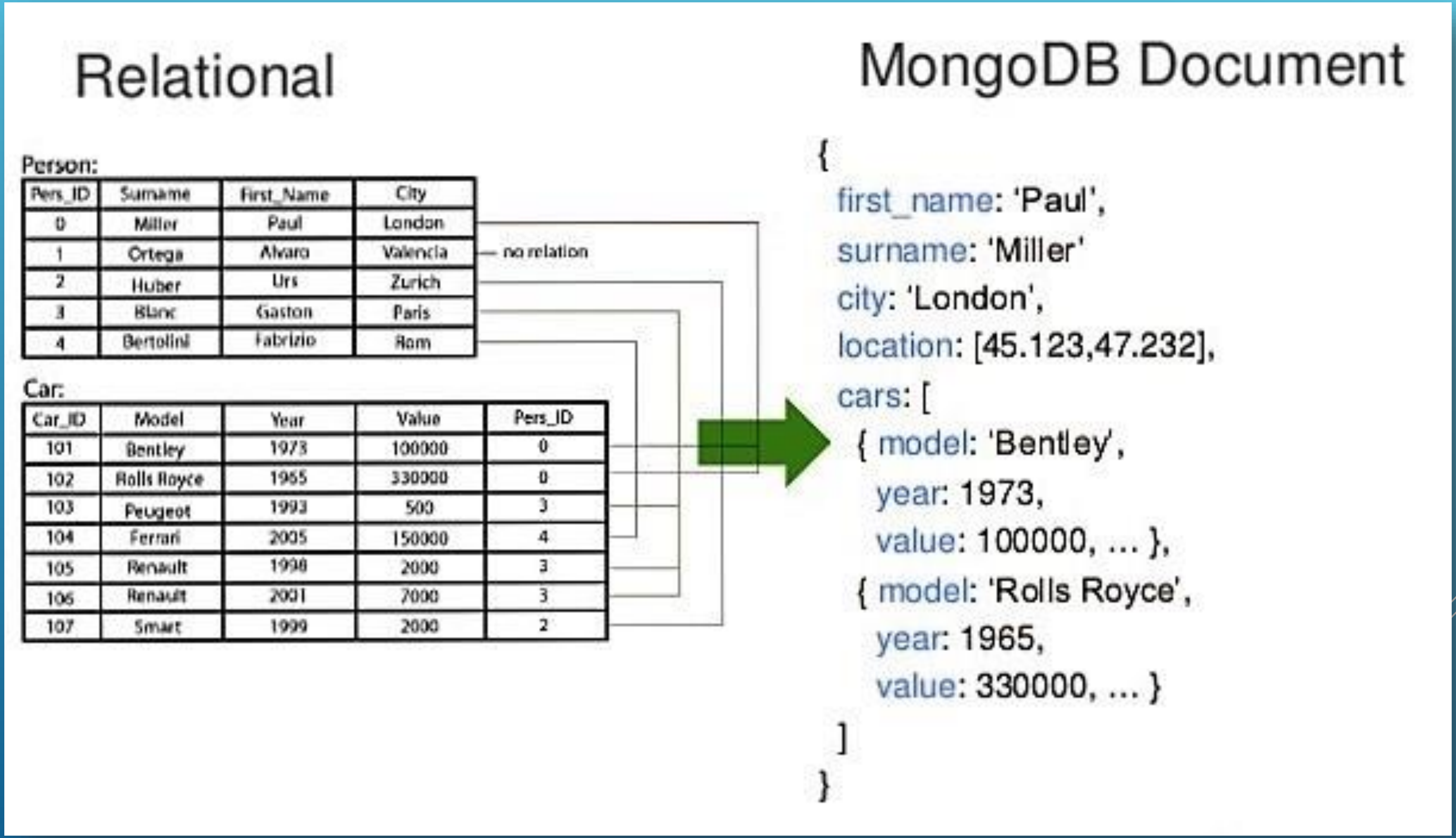
- Array and Embedded document : A field can hold this value type avoiding the use of joins

- BSON format : Docs are stored in database with this format to speed up read operations

- Horizontal scalability : It's possible to scale out the storage system in a distributed environment

MongoDB vs RDBMS

Comparison between table relationship and embedded documents



Relational

Person:

Pers_ID	Surname	First_Name	City
0	Miller	Paul	London
1	Ortega	Alvaro	Valencia
2	Huber	Urs	Zurich
3	Blanc	Gaston	Paris
4	Bertolini	Fabrizio	Rom

— no relation

Car:

Car_ID	Model	Year	Value	Pers_ID
101	Bentley	1973	100000	0
102	Rolls Royce	1965	330000	0
103	Peugeot	1993	500	3
104	Ferrari	2005	150000	4
105	Renault	1998	2000	3
106	Renault	2001	7000	3
107	Smart	1999	2000	2

MongoDB Document

```
{
  first_name: 'Paul',
  surname: 'Miller'
  city: 'London',
  location: [45.123,47.232],
  cars: [
    { model: 'Bentley',
      year: 1973,
      value: 100000, ... },
    { model: 'Rolls Royce',
      year: 1965,
      value: 330000, ... }
  ]
}
```

KASCADE Document Structure

(477) Document (4)	56d56420521811a6298ef7f9...	Document
_id	56d56420521811a6298ef7f9	ObjectId
general (7)		Document
Gt	894646292	Int32
Mt	261796200	Int32
P	1009.4093627929688	Double
T	22.789976119995117	Double
R	877	Int32
Ev	1862	Int32
Datetime	1998-05-08T16:51:32Z	DateTime
array (9)		Document
E	14.701221466064453	Double
Xc	-0.22888532280921936	Double
Yc	10.894941329956055	Double
Ne	4.654540061950836	Double
Nmu	3.6415719985961914	Double
Age	0.98474907875061035	Double
Ze	15.065551450424932	Double
Az	108.01087666653247	Double
Stations (252)		Array
calorimeter (2)		Document
Nhad	3	Int32
Ehad	11.880748748779297	Double

Station Key stores parameters for each detector as an array of embedded documents :

{ "_id" : ObjectId("56d56420521811a6298ef7f9"), "general" : { "Gt" : 894646292, "Mt" : 261796200, "P" :
1009.4093627929688, "T" : 22.789976119995117, "R" : 877, "Ev" : 1862, "Datetime" : ISODate("1998-05-
08T16:51:32Z") }, "array" : { "E" : 14.701221466064453, "Xc" : -0.22888532280921936, "Yc" :
10.894941329956055, "Ne" : 4.6545400619506836, "Nmu" : 3.6415719985961914, "Age" :
0.98474907875061035, "Ze" : 15.065551450424932, "Az" : 108.0108766653247, "Stations" : [{
"Distance" : 143.5, "Sid" : 1, "EDensity" : 0.0 }, { "Distance" : 135.5, "Sid" : 2, "EDensity" : 0.0 }, {
"Distance" : 125.5, "Sid" : 3, "EDensity" : 0.0 }, { "Distance" : 134.0, "Sid" : 4, "EDensity" : 0.0 }, {
"Distance" : 128.5, "Sid" : 5, "EDensity" : 0.0 }, { "Distance" : 122.0, "Sid" : 6, "EDensity" : 0.0 }, {
"Distance" : 110.5, "Sid" : 7, "EDensity" : 0.0 }, { "Distance" : 117.5, "Sid" : 8, "EDensity" : 0.0 }, {
"Distance" : 107.5, "Sid" : 9, "EDensity" : 0.0 }, { "Distance" : 99.5, "Sid" : 10, "EDensity" : 0.0 }, {
"Distance" : 89.5, "Sid" : 11, "EDensity" : 0.0 }, { "Distance" : 97.5, "Sid" : 12, "EDensity" : 0.0 }, {
"Distance" : 125.0, "Sid" : 13, "EDensity" : 0.0 }, { "Distance" : 116.0, "Sid" : 14, "EDensity" : 0.0 }, {
"Distance" : 107.0, "Sid" : 15, "EDensity" : 1.3183887004852295, "Arrival" : 337, "MDensity" : 0.0 }, {
"Distance" : 117.0, "Sid" : 16, "EDensity" : 0.0 }, { "Distance" : 117.0, "Sid" : 17, "EDensity" : 0.0 }, {
"Distance" : 112.5, "Sid" : 18, "EDensity" : 0.65919435024261475, "Arrival" : 295 }, { "Distance" : 100.0,
"Sid" : 19, "EDensity" : 0.0 }, { "Distance" : 105.0, "Sid" : 20, "EDensity" : 0.0, "MDensity" :
0.31962788105010986 }, { "Distance" : 110.0, "Sid" : 21, "EDensity" : 0.0 }, { "Distance" : 108.5, "Sid" : 22,
"EDensity" : 0.65919435024261475, "Arrival" : 271, "MDensity" : 0.0 }, { "Distance" : 95.5, "Sid" : 23,
"EDensity" : 0.0 }, { "Distance" : 97.0, "Sid" : 24, "EDensity" : 0.0 }, { "Distance" : 84.5, "Sid" : 25,
"EDensity" : 0.65919435024261475, "Arrival" : 288, "MDensity" : 0.0 }, { "Distance" : 82.5, "Sid" : 26,
"EDensity" : 0.0 }, { "Distance" : 69.5, "Sid" : 27, "EDensity" : 0.65919435024261475, "Arrival" : 275,
"MDensity" : 0.0 }, { "Distance" : 71.5, "Sid" : 28, "EDensity" : 1.3183887004852295, "Arrival" : 274,
"MDensity" : 0.0 }, { "Distance" : 93.0, "Sid" : 29, "EDensity" : 0.0 }, { "Distance" : 88.0, "Sid" : 30,
"EDensity" : 0.65919435024261475, "Arrival" : 315, "MDensity" : 0.0 }, { "Distance" : 76.0, "Sid" : 31,
"EDensity" : 0.0 }, { "Distance" : 82.0, "Sid" : 32, "EDensity" : 0.65919435024261475, "Arrival" : 290,
"MDensity" : 0.0 }, { "Distance" : 108.5, "Sid" : 33, "EDensity" : 0.0 }, { "Distance" : 109.5, "Sid" : 34,
"EDensity" : 0.0 }, { "Distance" : 97.0, "Sid" : 35, "EDensity" : 0.0 }, { "Distance" : 95.5, "Sid" : 36,
"EDensity" : 0.65919435024261475, "Arrival" : 262, "MDensity" : 0.0 }, { "Distance" : 112.5, "Sid" : 37,
"EDensity" : 0.65919435024261475, "Arrival" : 241, "MDensity" : 0.0 }, { "Distance" : 116.5, "Sid" : 38,

Aggregation Method

The Aggregation pipeline framework divides the workflow into a chain of stages that modify, delete or filter out documents processed by the previous phases.
Unlike one single query , this method would let in theory to split it in multiple stages in order to scan docs with different indexes, as showed in this example

$ match : query on E – Nmu → **E – Nmu** compound index

$ match : query on Age → **Age** single index

However, only the first stage takes advantage of index and the execution time doesn't benefit from more indexes; indeed, the splitting query penalizes significantly the performance.

Bibliography

- J. Wochele, D. Kang, D. Wochele, A. Haungs, S. Schoo, ***KCDC User Manual*** (www.kcdc.ikp.kit.edu) 15/03/15
- K. Chodorow, M. Dirolf, ***MongoDB: The Definitive Guide*** (O' Reilly), 09/10
- ***KCDC - The KASCADE Cosmic-ray Data Centre***, ECRS 2014 - Kiel, Germany; 1. - 5.9.2014
- MongoDB White paper, ***Top 5 Considerations When Evaluating NoSQL Databases***, 07/16
- ***The KASCADE Cosmic-ray Data Centre (KCDC)*** , ICRC 2015 - The Hague, Netherlands; 30.7. - 6.8.2015
- ***The KASCADE Cosmic ray Data Center - providing open access to astroparticle physics research data***, Helmholtz Open Access Webinars on Research Data Webinar 15; 8. - 12.11.2013
- ***KCDC - publishing research data from the KASCADE experiment***, Helmholtz Open Access Workshop, DESY, Hamburg; 11.6.2013