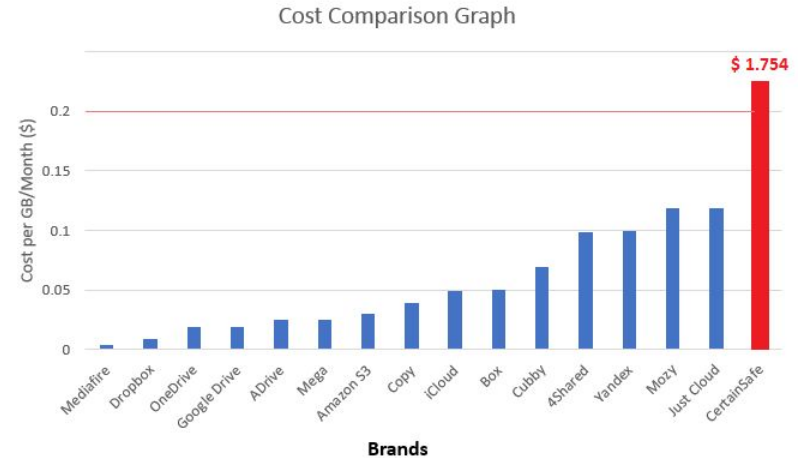
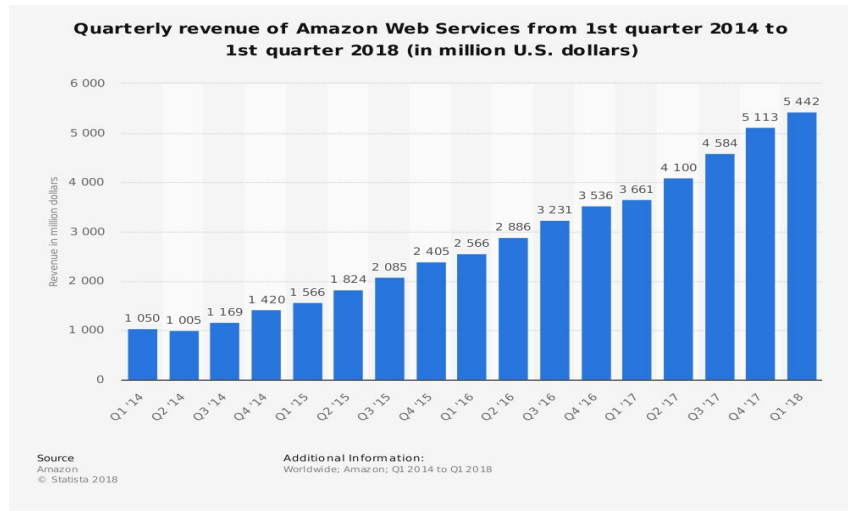
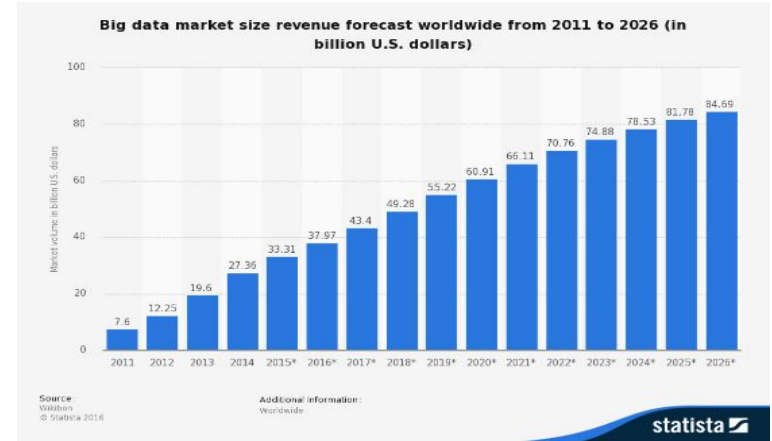


HSS

HIGH SPEED STREAMING

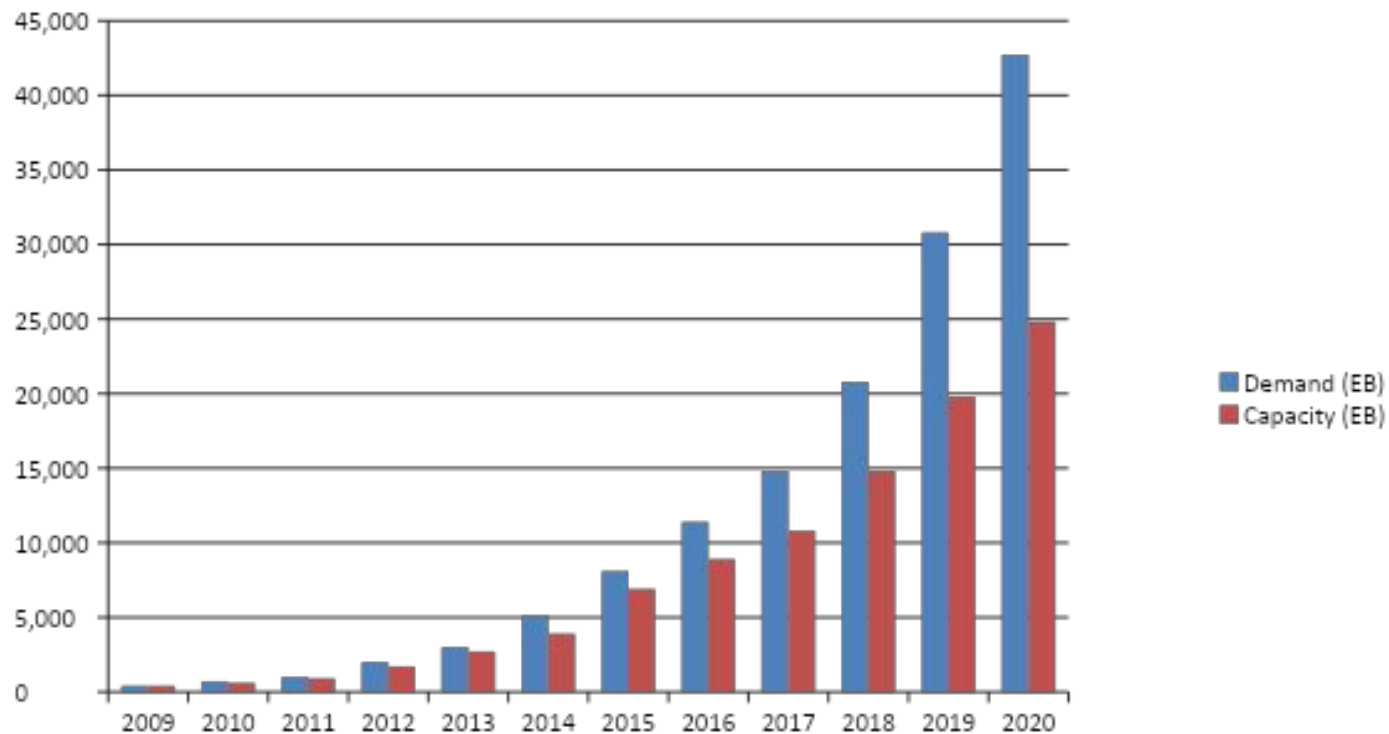
Big Data World



Big Data World

- Exponential data growth
 - Video, audio, image, multi-media, IOT, drones
 - 40 Zetabyte of data in 2020 doubling every year
 - data growth is faster than storage cost reduction
 - pure data storage cost
 - Network bandwidth/cost for transporting data
 - Databases are storing and handling hot data and high value data only – big amount is stored in Data Lakes
 - Cost factor of data storage is one of the main limiting factors for value extraction
- ⇒ There is a need for more effective Data Compression Algorithms
- In 2025 about 40% of all economic activities will be routed through one of the big cloud service and data providers
- >Control of the data is the key success factor for future economies

Data Storage Demand and Capacity



World of Data Compression

- Data compression methods differ in their basis. They are suitable for different types of data, and produce different results. In fact, they are all based on the same principle, i.e. Removing the redundant data.
- Hundreds of data compression methods can be classified under 4 main categories:
 1. Run Length Coding
 2. Statistical Methods
 3. Dictionary-Based Methods
 4. Transformation

Image Compression World

Image Compression can be made in two ways: Lossless Compression and Loosy Compression.

- Lossless Image Compression Methods: RLE, JPEG-LS, JBIG Standard, JBIG-2
- Image File Formats using Lossless Compression: GIF, TIFF, PNG
- Loosy Image Compression Methods: JPEG Standard, JPEG2000 Standard
- Loossles Motion Video Compression Methods: Huffvuv, MSU (Moscow State University), SheerVideo, CorePNG, MSU Lossless Video Codec, LCL, Lagarith, SKYUV [Smith, 2000], Lossless Motion JPEG
- Loosy Motion Video Compression Methods: ITU-T H.261, ITU-T H.263, MPEG1, MPEG2, MPEG4, AVC (ITU-T H.264, ISO/IEC MPEG-4 Part 10)

HSS Coding and Compression System

- Unmatched compression rate
- High compression rate even for „uncompressable data“ (compression just with loss) – i.e. 1GB VOB extension to 80MB
- Compression rate about 92%
- Compressed format can be streamed (doesnt have to be unpacked before used)
- System has its own cryptology.
- System has its own platform and programming language.
- System has secured structure called Shield Technology – an own anti-virus and firewall.

HSS Coding and Compression System

FILE QUALITY	SIZE	PERCENTAGE OF COMPRESSION	HSS FILE	MACRO BLOCK VOLUME	ENTROPY CODING	COMPRESSION TIME	OPENING TIME
MPEG-1	1 GB	%92.4	76 Mb	16x16	HUFFMAN	0.64 sn	0.65 sn
MPEG-2	1 GB	%92.8	72 Mb	16x16 (Interlaced browsing)	HUFFMAN	0.64 sn	0.65 sn
MPEG-4 (Part 2) Visual H.263	1 GB	%92.5	75 Mb	16x16	HUFFMAN	0.64 sn	0.65 sn
MPEG-4 (Part 10) AVC H.264	1 GB	%93.74	62,6 Mb	16x16	Exp-Golomb, CAVLC,CABAC	0.64 sn	0.65 sn

HSS Fields of Usage

- Data storage and saving
- GSM networks
- Artificial intelligence applications (data streaming)
- Internet TV broadcasting, TV broadcasting, all sorts of multi-media applications, cloud storage solutions
- Remote sensing technologies
- Unmanned transport technologies

Test Results

Original file: Ubuntu VM image, 4244 MB. Short table showing only best methods for each compressor.

Archiver	Mode	Compressed size, MB	Compression time, sec	Decompression time, sec
Zip 3.0	-9	1398	464	21
BZip2 1.0.5	-9	1279	637	186
RAR 3.90 [64]	Normal	1119	490	37
InnoSetup	Ultra	987	1329	73
7-zip 9.07 [64]	Ultra 256mb	945	1417	86
FreeArc 0.60	Ultra	938	1560	85
NanoZip 0.07 [64]	-cO	909	4692	667
NanoZip 0.07 [64]	-cc	849	6375	2574
REDA HSS	Abz	344	880	243

Test results

Compressed file recompressed by HSS

Archiver	Mode	Compressed size, MB	Compressed file recompressed by HSS	Compression time, sec	Decompression time, sec
Zip 3.0	Abz	1398	125,82	280	80
BZip2 1.0.5	Abz	1279	115,11	265	70
RAR 3.90 [64]	Abz	1119	100,71	232	60
InnoSetup	Abz	987	88,83	204	56
7-zip 9.07 [64]	Abz	945	85,05	195	54
FreeArc 0.60	Abz	938	84,42	193	53
NanoZip 0.07 [64]	Abz	909	81,81	185	52
NanoZip 0.07 [64]	Abz	849	76,41	176	48

Arçelik Test Results

Filename	SHA-256	File Size	Extension	Time1 compress	Time2 decompress	Compressed Size	Rate
test.img	661db18b3422ce1642147349f66e7d13552630b134c202e857796deccfe3a4f2	1.073.741.824	img	1dk 1sn	11 sn	83205633	92,25087157
film	cd87884a50ac9545af071353858175ed3b815974accc625565370980ace7415d	2.428.767.917	mp4	31 sn	22 sn	165782775	93,1742027
ubuntu1404t.zip	5b2dbef56c3a8aa8327f82cbd1569ed32f81acd76fae389168a9dcb9a6522083	1.003.094.138	zip	1 dk 1 sn	9 sn	73321208	92,69049581
TR300_V01.07.14.zip	a9941f106c2216d3824549418557dd139ce30b88ced4333bdb33a2a521001b5a	49.747.029	zip	12 sn	12 sn	3634960	92,69311138
test2.img	afecb6f478c7eac8cfa85a2d9b6cd84d51b3ec519cea9e70cba17faf175d9da0	1.000.000.000	img	0	0	83205600	91,67944

Integration in cloud storage

The diagram illustrates a three-tier architecture for cloud storage integration. At the top is a blue rounded rectangle labeled 'applications'. Below it is a grey rectangle labeled 'HSS access compression layer'. At the bottom is a blue cloud shape labeled 'Amazon data lake (S3)'. The components are stacked vertically, indicating a flow from applications through the compression layer to the data lake.

applications

HSS access compression layer

Amazon data lake (S3)

HSS compression layer (HSSCL) is implementing the S3 API)

Applications can access either S3 native or via HSSCL seamless

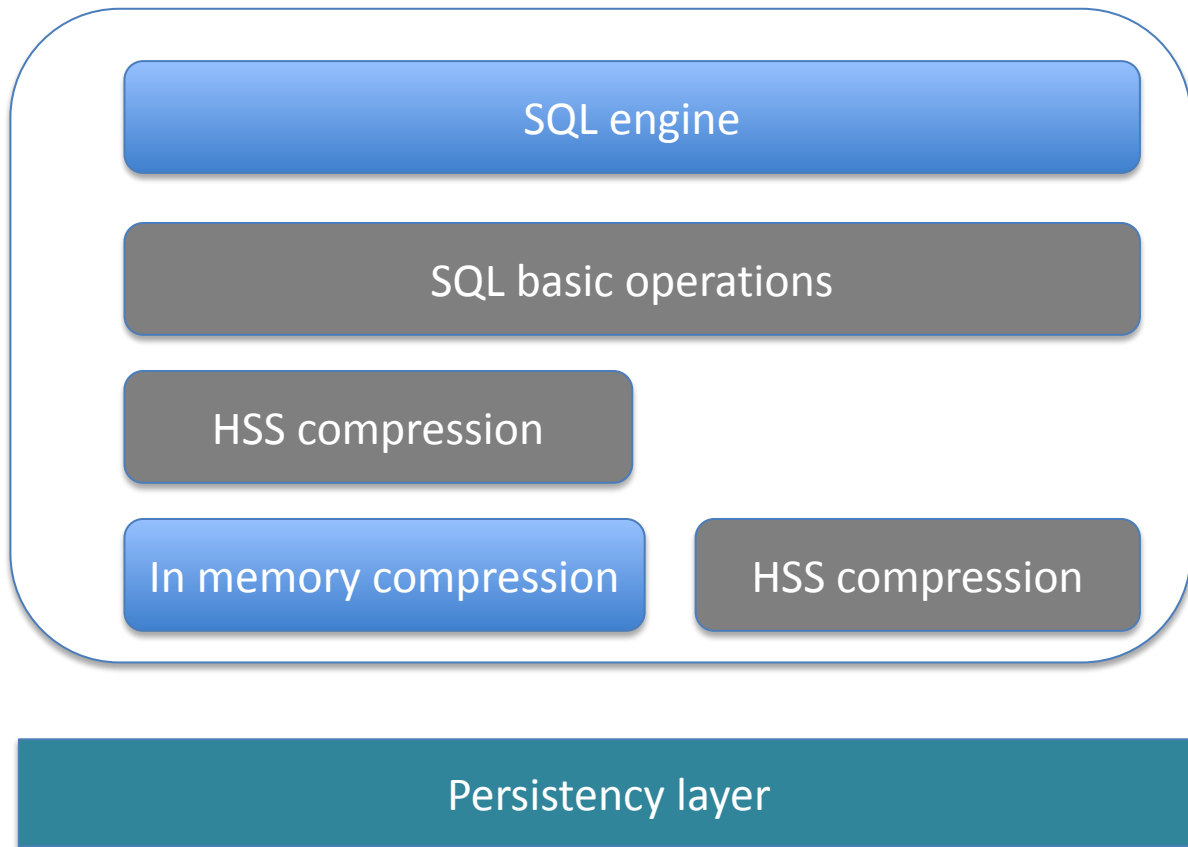
HSSCL is able to offer 1/2 of the Storage costs as S3 native with the Same guarantees

Consumption model : we provide HSSCL service on Amazon (and Other cloud providers)

Risk :

- Stability of the service
- compression costs

Integration into in memory databases



HSS is a library which is used either as a new compression or a Second level compression

Basic SQL operations will be Implemented on top of HSS

HSS is licensed to in memory Database providers

In memory providers will save Massive costs in DBaaS deployments

Risk :

- License model
- IP

Tool integration

Application specific stream consumers

Consume compressed data
Without decompressing the whole
Data – just piecewise

Implement streaming consumers
Like video streaming or table
Readers, parquet/orc readers ...

Piecewise read (stream)

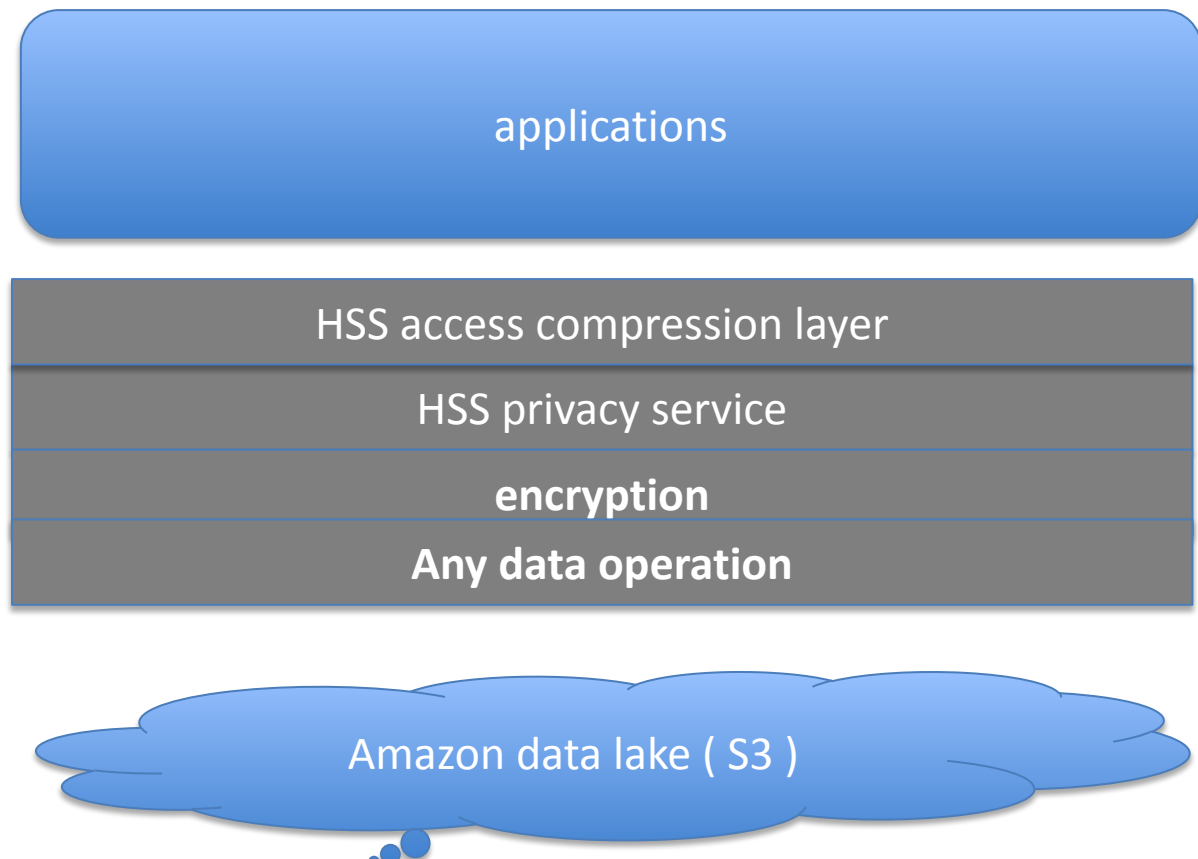
Allows to reduce memory usage
By investing more CPU

Compressed data

Risk :

- Quite some effort cause of tool variety
- Distribution of tools

Vision : provide additional data services during compression



Enhance the compression service
With other data services

As data are anyhow converted,
And streamed through the service
additional functionality can be
Offered

Consumption model : we provide
extended services on Amazon (and
other cloud providers)

Demo

Thanks