

DESILO

SQL-Based Data Analysis Platform Using Homomorphic Encryption

Jae Lee, Donghoon Yoo, Joohyung Lee

Shaping the future of data collaboration

Privacy-Preserving Applications
PET Full-Stack Optimization
Cryptography Libraries

CAGR

23%

64.1 ZB generated in 2020. **New data** is being created and collected at an exponential rate

CAGR

30.4%

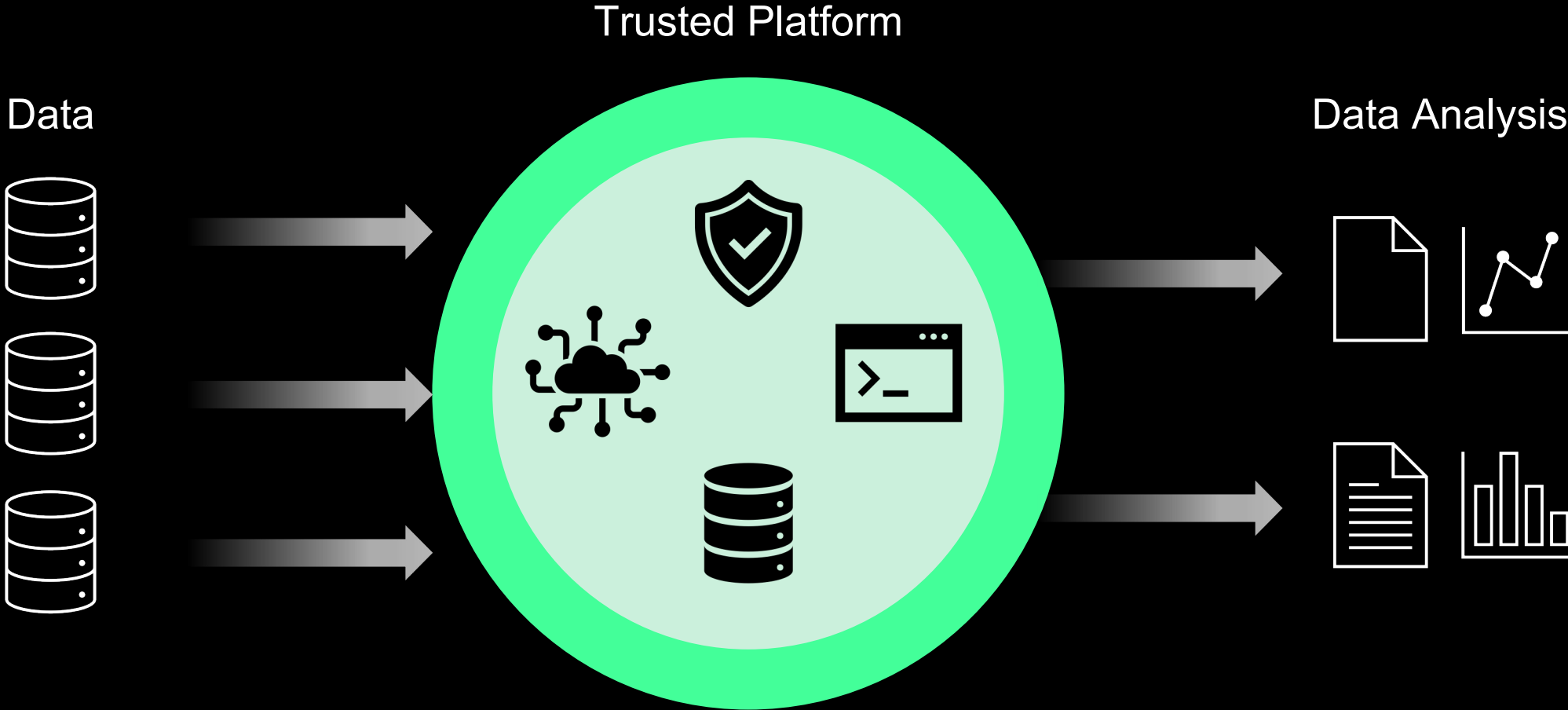
\$ 31.7B **data analytics** market in 2021. The need for extracting insight from data is exploding

CAGR

4.5%

\$ 257.2B **data broker** market in 2021. Not enough data is being circulated, shared, and consumed



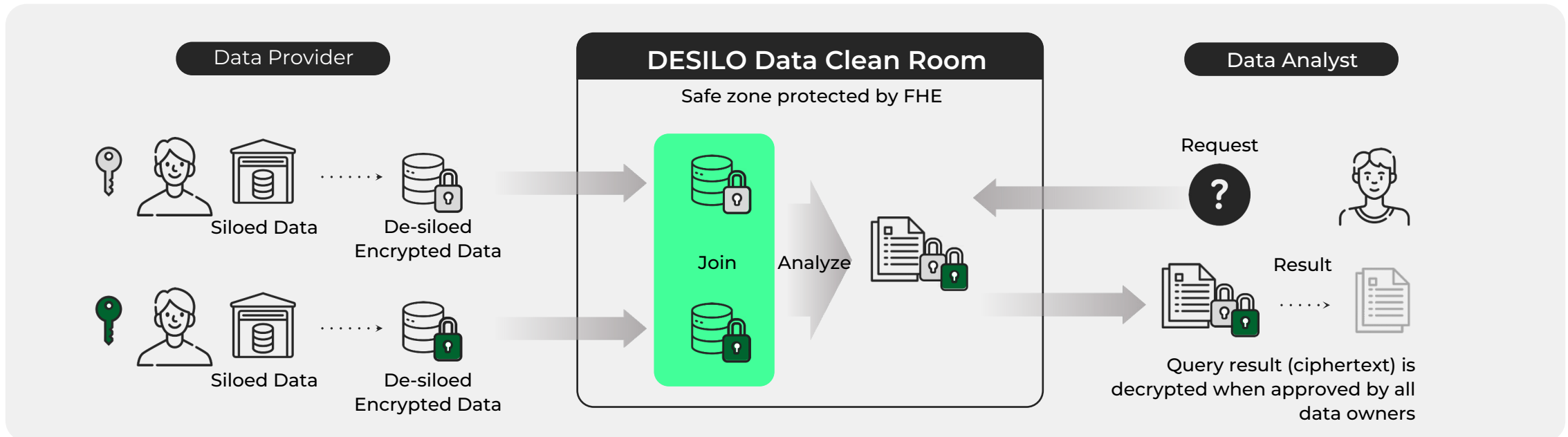


Relies on credibility of platform

Not scalable for true DaaS or as data exchange

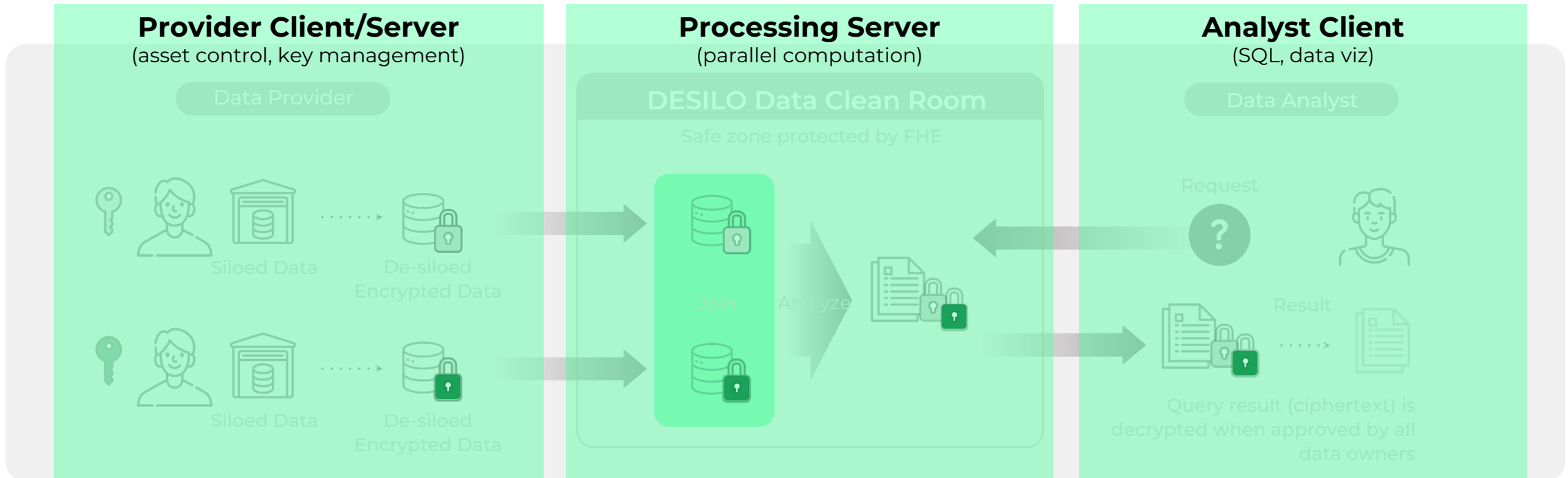
DESILO Data Clean Room

- Built-in protection against data leakages & breaches
- Redefines DaaS, enabling a path towards a secure and open data ecosystem



DESILO Data Clean Room

- Built-in protection against data leakages & breaches
- Redefines DaaS, enabling a path towards a secure and open data ecosystem



Data Providers

- Effortless data asset management
- Rigorous data governance & security
- Data monetization with control

Data Analysts

- Secure by design
- User-friendly SQL queries
- Seamless access to data

Data Providers

- Effortless data asset management

The screenshot displays the 'Data Assets' management interface. On the left, a sidebar contains 'Data Assets' and 'Query Approval'. The main area shows a list of data assets with columns for name, description, and timestamp. An 'Upload data files' modal is open, providing instructions for uploading CSV files and showing a progress bar for 'accountbook-sampled_clean.csv' at 11%.

Data Assets

Data Assets	Timestamp
KCD [교육연구용] 한국신용데이터 transaction	2024-01-29 16:48
health-info [교육연구용]뱅크샐러드 샘플 유저 건강검진 결과	2024-01-22 20:39
asset-card [교육연구용]뱅크샐러드 샘플 유저 카드 정보	2024-01-22 20:37
asset-insurance [교육연구용]뱅크샐러드 샘플 유저 보험 정보	2024-01-22 20:35
asset-invest [교육연구용]뱅크샐러드 샘플 유저 투자자산 정보	2024-01-22 20:29
asset-bank [교육연구용]뱅크샐러드 샘플 유저 은행 계좌 정보	2024-01-22 20:26
transaction-info [교육연구용]뱅크샐러드 샘플 유저 카드결제 내역	2024-01-22 20:21
user-info [교육연구용]뱅크샐러드 샘플 유저 정보	2024-01-22 20:14

Upload data files

Click here to browse or drag and drop your CSV file.
File extension: CSV | Encoding format: utf-8

* Keep your browser open until the upload finishes. The upload will not be completed correctly if the browser is closed.
* Click an uploaded file to register the data asset.

Upload list

File Name	Progress	Timestamp
accountbook-sampled_clean.csv	11%	2024-01-22 20:29

provider1
provider1@gmail.com

KEY BENEFITS

Data Providers

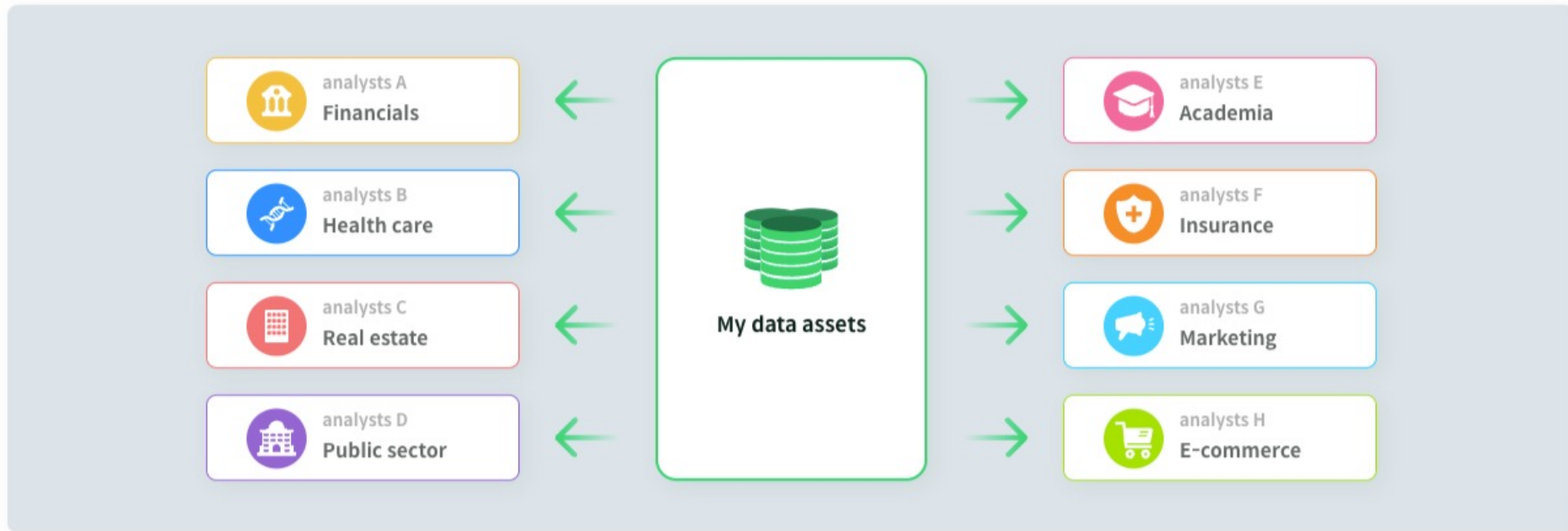
- Rigorous data governance & security

The screenshot shows the 'Edit Data Asset' interface. On the left is a dark sidebar with 'DATA CLEAN ROOM FOR PROVIDERS' at the top, 'Data Assets' (selected), and 'Query Approval'. The main area has a title 'Edit Data Asset' and 'Cancel'/'Save' buttons. It features several sections: 'Auto Approve' (checked), 'Minimum aggregation threshold' (set to 20, highlighted with a green dashed box), and 'Check column name and type' (with a search bar). Below is a table with columns: Original Name, Column Name, Data Type, Description, and Matching Key Only (highlighted with a green dashed box). A 'Uploading data files' notification shows 'sample.csv' being uploaded.

Original Name	Column Name	Data Type	Description	Matching Key Only
_month	<input type="text" value="_month"/>	<input type="text" value="int"/>	<input type="text" value="Description"/>	<input type="checkbox"/>
_quarter	<input type="text" value="_quarter"/>	<input type="text" value="int"/>	<input type="text" value="Description"/>	<input type="checkbox"/>
_year	<input type="text" value="_year"/>	<input type="text" value="int"/>	<input type="text" value="Description"/>	<input type="checkbox"/>
age	<input type="text" value="age"/>	<input type="text" value="string"/>	<input type="text" value="Description"/>	<input type="checkbox"/>
b_id	<input type="text" value="b_id"/>	<input type="text" value="int"/>	<input type="text" value="Description"/>	<input checked="" type="checkbox"/>
business_square_size	<input type="text" value="business_square_size"/>	<input type="text" value="float"/>	<input type="text" value="Description"/>	<input type="checkbox"/>
duration	<input type="text" value="duration"/>	<input type="text" value="float"/>	<input type="text" value="Description"/>	<input type="checkbox"/>

Data Providers

- Data monetization with control



Data Analysts

- Secure by design

The screenshot displays the 'Data Spaces' section of the DESILO interface. On the left, a sidebar shows 'DATA CLEAN ROOM FOR ANALYSTS' with navigation options for 'Projects' and 'Data Spaces'. The main content area is titled 'Data Spaces' and shows a list of 'All data assets' with one asset selected: 'analyst1 + analyst2 + provider1'. The 'Data Asset Detail' view for 'KCD' is shown, including a search bar and a table of asset information. A red dashed box highlights the 'Minimum aggregation threshold' set to 20. Below this, the 'Columns' section lists 'b_id' (int) and 'location_sido' (string), with their respective data types and values highlighted by red dashed boxes.

Data Asset Detail

Search by column name

Data Asset Information Data Provider provider1@gmail.com

Data asset name KCD	Row count 662135	Minimum aggregation threshold 20	Created at 2024-01-29 16:48
Public description [교육연구용] 한국신용데이터 transaction			

Columns

All types String Int Float Date

int b_id			
Not Null / Null Values			
662135 / 0			
string location_sido			
Not Null / Null Values	Most Frequent Value (Mode)	Count of Mode	
662135 / 0	경기도	368634	

Data Analysts

- User-friendly SQL queries

DATA CLEAN ROOM FOR ANALYSTS

← Create workflow Start Workflow

Name *

Description

SQL Query Setting Form Editor Analysis Type SQL

Please write a query for the workflow to use.

SELECT COUNT(*) + New Clause

FROM provider1.user-info + New Clause

COMPOUND COLUMN + New Clause

JOIN + New Clause

WHERE + New Clause

GROUP BY + New Clause

ORDER BY + New Clause

HAVING + New Clause

LIMIT + New Clause

* Because homomorphic encryption includes some level of approximation, the computed results may include an error of up to 0.1% compared to the results computed without homomorphic encryption.

AVAILABLE KEYWORDS

SELECT ^

AVERAGE

COUNT

COVARIANCE_POPULATION

COVARIANCE_SAMPLE

PEARSON

STANDARD_DEVIATION_POPULATION

STANDARD_DEVIATION_SAMPLE

STANDARD_ERROR_OF_THE_MEAN

SUM

T_TEST

VARIANCE_POPULATION

VARIANCE_SAMPLE

FROM

JOIN

WHERE

GROUP BY

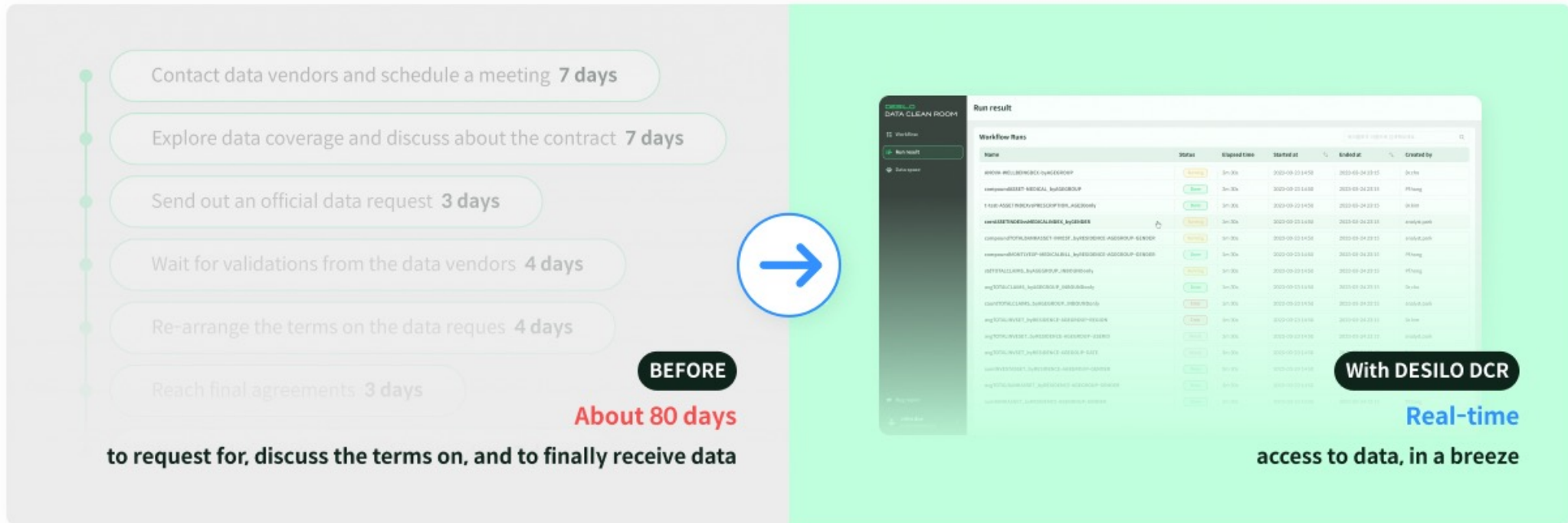
ORDER BY

HAVING

LIMIT

Data Analysts

- Seamless access to data



* Above illustrates an exemplary case based on data collaboration processes in general; the actual duration of each stage may vary by case.

Data Representation

- Limit input size
- Normalize data

Performance

- Parallelize computation
- Re-order and distribute operators

Data Governance

- Automate/adopt query alert & approval
- Enforce minimum aggregation threshold

Liberate.FHE

- Up to 200+ times faster, 3+ times code productivity, near-perfect accuracy (e.g. vs. MS SEAL)
- Native CUDA support, multi-GPU scaling, optimized GPU partitioning, CPU interoperability

```

1 sk_cpu = engine.cpu(sk)
2 sk_cuda = engine.cuda(sk_cpu)
3
4 galk_cpu = engine.cpu(galk)
5 galk_cuda = engine.cuda(galk_cpu)
6
7 ct_cpu = engine.cpu(ct)
8 ct_cuda = engine.cuda(ct_cpu)
9

```



	SEAL				LFHE	speedup
	mult	relin	rescale	total	total	
Silver	11.6	186	28.9	226.5	7.74	29.26
Gold	48.7	1570	133	1751.7	17.7	98.97
Platinum	212	15400	587	16199	73.7	219.80

* Silver: logN=15, special primes=2, gpus=1
* Gold: logN=16, special primes=4, gpus=2
* Platinum: logN=17, special primes=6, gpus=2

* 단위: ms

Liberate.FHE

<https://github.com/Desilo/liberate-fhe>
<https://docs.desilo.ai/liberate-fhe>

Poster

The screenshot shows the GitHub repository for Liberate.FHE. The main content area displays the 'Homomorphic Statistics' tutorial. The tutorial title is 'Homomorphic Statistics' and it describes the purpose of the tutorial: to calculate the mean and variance for all the data using approximately 10 million encrypted records. The tutorial includes a 'Getting Started' section with 'Quick Start' and 'Installation' instructions. A code block shows the beginning of a Python script named 'var.py'.

```

import numpy as np
from liberate import fhe
from liberate.fhe import presets

# generate engine
params = presets.params["gold"]
engine = fhe.ckks_engine(verbose=True, **params)

# generate keys
sk = engine.create_secret_key()
pk = engine.create_public_key(sk)
evk = engine.create_evk(sk)
gk = engine.create_galois_key(sk)

# set num of data
target_N = 10 ** 7 # 10,000,000
required_num_ct = round(target_N / engine.num_slots)

# generate Data
    
```

The poster is titled 'Liberate.FHE: A New FHE Library for Bridging the Gap Between Theory and Practice with a Focus on Performance and Accuracy'. It is authored by Juwhan Kim, Hanyul Ryu, and Donghoon Yoo. The poster is divided into several sections: Abstract, Liberate.FHE Design Philosophy, Results and Example Codes, and Conclusions. It also includes a table of HE Operations and a QR code for the documentation website.

Abstract
 A new implementation of the Fully Homomorphic Encryption (FHE) library, namely Liberate.FHE, is presented. In terms of performance and accuracy, despite significant advancements in previous works, the adoption of modern computing systems, such as GPU acceleration, has been sparse, and the accuracy of the homomorphic computations could have been better than the theoretical conditions. This work addresses the two optimal issues by integrating the existing computing system and engineering. Since the FHE needs to be implemented in the real world, Liberate.FHE supports multi-GPU operations naturally, and its application programming interface (API) is required to be simple in engineering the real world and presents. The main idea behind the design decision is that non-operations can be used by the library to be easily readable and integrated with more extensive software frameworks. The resulting implementation can be accessed at <https://github.com/Desilo/liberate-fhe>.

Major Algorithms in Liberate.FHE
New Chained Prime Selection Method
 Liberate.FHE introduces an error-free sampling method through a novel chained prime selection approach, leveraging the distinct but related primes to mitigate recurring errors inherent in RNS-based FHE systems which use a single parameter q , and a modulus $q = \prod_{i=1}^L q_i$. The proposed method, detailed in Equation 1, optimizes the selection of prime primes to minimize cumulative deviation risk, crucial for maintaining the accuracy of homomorphic operations with N . It employs a strategic search for prime primes that support the desired modulus, effectively T1 adjusting the direction of the search to balance cumulative deviations. This process iterates, combined with a pre-walking strategy that identifies other candidates before finalization, significantly enhances the accuracy and efficiency of Liberate.FHE's homomorphic computations.

Encoding/Decoding using Mega-cyclic FFT
 Our existing method involves recursive candidate numbers and conversion to real numbers, with a focus on minimal accuracy loss. It employs a novel mega-cyclic FFT, transforming complex errors for error cancellation. This approach uses a combinatorial search to match optimal candidates, integrating recently proposed algorithms without extra scaling, simplifying encryption while maintaining precision and safety.

Division by the Chained Modulus and Reduction
 In the context of Fully Homomorphic Encryption (FHE), a novel method for modular division by a chained prime is introduced. All operations traditionally used during modulus switching. This method introduces modular division by multiple chained primes, reducing the number of operations. The proposed method, detailed in Equation 2, optimizes the accuracy of FHE systems by reducing the number of operations.

Key Switching
 Liberate.FHE utilizes a unique key switching approach, inspired by a novel method [22] focusing on direct operations on the FHE key matrices rather than applying the conventional scheme decomposition technique. The method involves transformed key components, and only through a process where a threshold from vector h is reached, with a constraint that key switching should be performed only when the threshold is reached. This direct approach simplifies the key switching process, leading to more efficient key switching during the encryption process.

RNS Base Extension
 The decomposition approach for optimizing RNS representation of integers involves a three-part process:
 • Integer focusing on generating a set of real parameters for efficiency.
 • Followed by the substitution of a close-order using post-processed values.
 • Concluding with a back extension to integrate uniform values. This method leverages the 'join' function for integer operations, but within R_{q_i} , demonstrating its applicability beyond the initial partition to incorporate arbitrary base partitions, overcoming common problems in homomorphic operations.

Abstract
 Liberate.FHE attempts to strengthen the computations by extending multi-GPU. In this work, homomorphic encryption (FHE) is implemented across and across abstract construction. Additionally, several design decisions were made to meet the usability of the distributed system:
 • Make the number of operations constant.
 • Make the software easily hackable.
 • Set the number of multi-GPU as the default.
 • Make the existing library easily integrated with the existing software, especially the artificial intelligence (AI) related ones.
 The above decisions led to building the library on top of the popular software package PyTorch. In Liberate.FHE, all the key and GPU flow are a collection of Python tensors. In that way, the FHE operations and the existing tensor flow can be shared and hacked easily. Also, the very low-level functions such as the homomorphic reduction and the related operations, Number Theoretic Transform (NTT), and Cryptographically Secure Pseudo Random Number Generators (CSPRNG) are built using the NVIDIA CUDA programming language. The low-level functions are then abstracted by the higher-level tensor functions using the C-API provided by PyTorch.

Practical Advancements
Data Partitioning over Multiple GPUs
 Liberate.FHE supports the very efficient data processing across multi-GPU setups, utilizing a tensor sharding strategy that minimizes data movement. Before computational load, and balancing distribution, the library provides a simple design for efficient, high-performance homomorphic encryption. The sharding strategy is designed to satisfy the following conditions:
 • A low sharding partition always resides in the same device.
 • The base channel is just into the master GPU.
 • Special prime classes are replicated at all GPUs.
 • The partitions are placed in GPUs in an evaluation of one partition will start the computation of the next partition at a different GPU.

Exact Reading in Reading
 Our study highlights that the accuracy of reading results in FHE is significantly influenced by the reading method employed. By introducing a reading procedure, $a = \frac{1}{m} \sum_{i=1}^m x_i$, and incorporating it into the reading process, we achieve more precise solutions, enhancing computational accuracy substantially.

Equation

$$\left(\frac{a_i - \bar{a}}{\sigma} \right) \sim \mathcal{N}(0, 1) \quad (1)$$

Heterogeneous Reduction Based Integer Arithmetic
 Liberate.FHE enhanced its capabilities with an optimized most-generous reduction, tailored for GPUs by using bitwise operations to perform modulus reduction and reduction without exceeding 64-bit store or wasting memory, significantly increasing computational efficiency.

Cryptographically Secure Pseudo Random Number Generator (CSPRNG)
 The new CSPRNG for Liberate.FHE, designed for enhanced security and efficiency in homomorphic encryption features:
 • CHACHA20S algorithm base for cryptographic robustness.
 • GPU parallelization via index-based generation for high performance (32).
 • Branchless binary tree search, optimizing GPU operations by avoiding conditional logic.
 • This streamlined approach addresses security and computational efficiency concerns associated with traditional CSPRNGs.

Military
 Liberate.FHE advances FHE with an 'off' content model, integrating distributed tasks and privacy (4). It offers high efficiency and dynamic parameter adaptability via individual secret keys and a shared public key, supported by our friendly FHE for collaboration, real-world applications.

Results and Example Codes
Results
 Liberate.FHE sets a new benchmark in FHE with key requirements for unparalleled accuracy and speed. It integrates innovative operations via Montgomery Reduction, avoiding floating-point errors, and optimizes the number of operations for each read/write operation. Its novel features, before introduction, strongly address the performance, transparency, cyclic rotation and integrity, with unique formulas mitigating module and division errors, vastly improving accuracy in homomorphic operations.

Table 1: Liberate.FHE benchmark (1e6)

Operation	Secret Key	Public Key
Key Creation	0.330	0.202
	0.255	0.255
	0.274	0.274
HE Operations	11.246	11.246
	4.150	4.150
	18.000	18.000
Round Trip	5.623	5.623
	1.051	1.051

Example Codes

```

from liberate import fhe
from liberate.fhe import presets

# generate engine
params = presets.params["gold"]
engine = fhe.ckks_engine(verbose=True, **params)

# generate keys
sk = engine.create_secret_key()
pk = engine.create_public_key(sk)
evk = engine.create_evk(sk)
gk = engine.create_galois_key(sk)

# set num of data
target_N = 10 ** 7 # 10,000,000
required_num_ct = round(target_N / engine.num_slots)

# generate Data
    
```

Conclusions
 Liberate.FHE bridges the gap between theoretical FHE and practical applications by providing a user-friendly, high-performance, and accurate library. Its novel design leverages modern computing resources for GPU and leverages GPU for non-operations, enhancing reading wider adoption and real-world applications of FHE. Further research directions include investigating homomorphic on multiple GPUs, exploring secure switching between FHE, and developing flexible and scalable multi-party FHE protocols on the FHE.

Features in Delivery
 • CKKS bootstrapping
 • Multi-party bootstrapping
 • Liberate.FHE GPU support
 • RNS-GPU-Disability for CKKS

References
 [1] DESILO, "Towards the New Homomorphic Encryption (FHE) Library, namely Liberate.FHE, is presented. In terms of performance and accuracy, despite significant advancements in previous works, the adoption of modern computing systems, such as GPU acceleration, has been sparse, and the accuracy of the homomorphic computations could have been better than the theoretical conditions. This work addresses the two optimal issues by integrating the existing computing system and engineering. Since the FHE needs to be implemented in the real world, Liberate.FHE supports multi-GPU operations naturally, and its application programming interface (API) is required to be simple in engineering the real world and presents. The main idea behind the design decision is that non-operations can be used by the library to be easily readable and integrated with more extensive software frameworks. The resulting implementation can be accessed at <https://github.com/Desilo/liberate-fhe>." (2024), arXiv:2408.12345.
 [2] DESILO, "Towards the New Homomorphic Encryption (FHE) Library, namely Liberate.FHE, is presented. In terms of performance and accuracy, despite significant advancements in previous works, the adoption of modern computing systems, such as GPU acceleration, has been sparse, and the accuracy of the homomorphic computations could have been better than the theoretical conditions. This work addresses the two optimal issues by integrating the existing computing system and engineering. Since the FHE needs to be implemented in the real world, Liberate.FHE supports multi-GPU operations naturally, and its application programming interface (API) is required to be simple in engineering the real world and presents. The main idea behind the design decision is that non-operations can be used by the library to be easily readable and integrated with more extensive software frameworks. The resulting implementation can be accessed at <https://github.com/Desilo/liberate-fhe>." (2024), arXiv:2408.12345.
 [3] DESILO, "Towards the New Homomorphic Encryption (FHE) Library, namely Liberate.FHE, is presented. In terms of performance and accuracy, despite significant advancements in previous works, the adoption of modern computing systems, such as GPU acceleration, has been sparse, and the accuracy of the homomorphic computations could have been better than the theoretical conditions. This work addresses the two optimal issues by integrating the existing computing system and engineering. Since the FHE needs to be implemented in the real world, Liberate.FHE supports multi-GPU operations naturally, and its application programming interface (API) is required to be simple in engineering the real world and presents. The main idea behind the design decision is that non-operations can be used by the library to be easily readable and integrated with more extensive software frameworks. The resulting implementation can be accessed at <https://github.com/Desilo/liberate-fhe>." (2024), arXiv:2408.12345.
 [4] DESILO, "Towards the New Homomorphic Encryption (FHE) Library, namely Liberate.FHE, is presented. In terms of performance and accuracy, despite significant advancements in previous works, the adoption of modern computing systems, such as GPU acceleration, has been sparse, and the accuracy of the homomorphic computations could have been better than the theoretical conditions. This work addresses the two optimal issues by integrating the existing computing system and engineering. Since the FHE needs to be implemented in the real world, Liberate.FHE supports multi-GPU operations naturally, and its application programming interface (API) is required to be simple in engineering the real world and presents. The main idea behind the design decision is that non-operations can be used by the library to be easily readable and integrated with more extensive software frameworks. The resulting implementation can be accessed at <https://github.com/Desilo/liberate-fhe>." (2024), arXiv:2408.12345.

Liberate.FHE Documentation Website

 Scan the QR code

PIPC approved Desilo Data Clean Room as providing adequate protection of personal data.

2024/01/25



개인정보보호위원회
Personal Information
Protection Commission



DEMO

2023

Co-hosted data science contest

- Pilot launch with real data
- 144 participants/teams
- Demonstrated reliable service (maximum daily processing capacity exceeds 1000 queries)



2024

Healthcare

Government

Private (Fin/Ads/Mfg)

Closing Remarks

DESILO Data Clean Room is an FHE-based platform that facilitates secure data flow & collaboration

Continued innovation in computation and data governance is essential to reach full market potential

DESILO

THANK YOU

Jae Lee

kyeongjae.lee@desilo.ai

<https://www.linkedin.com/in/kjaelee/>