Buliding a Structurally-Encrypted Relational Database

Zheguang Zhao

Brown University
zheguang.zhao@gmail.com

November 11, 2020

Outsourcing Data to the Cloud

Cloud services have become very common

Outsourcing Data to the Cloud

Cloud services have become very common

- Email, file hosting, collaborative document editing, data processing, backups and more.

Outsourcing Data to the Cloud

Cloud services have become very common

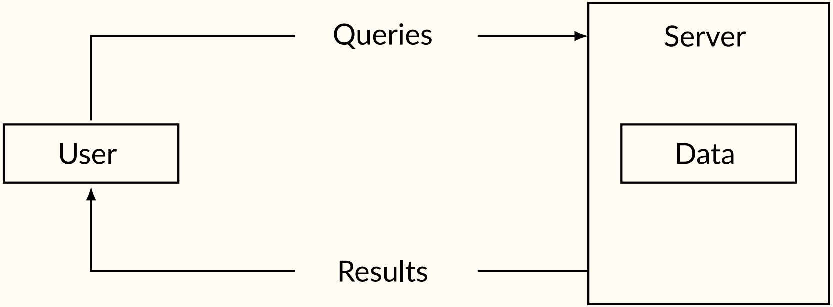
- Email, file hosting, collaborative document editing, data processing, backups and more.
- Outsourced infrastructure; Uptime, computation and storage on-demand.
- By 2020, 80% small & medium businesses in US will use cloud services [Col15].

Outsourcing Data to the Cloud

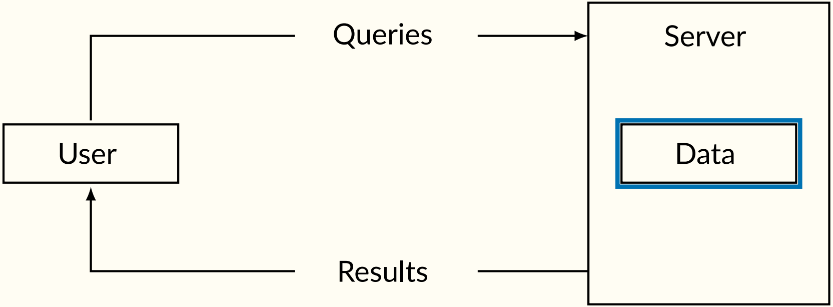
Cloud services have become very common

- Email, file hosting, collaborative document editing, data processing, backups and more.
- Outsourced infrastructure; Uptime, computation and storage on-demand.
- By 2020, 80% small & medium businesses in US will use cloud services [Col15].
- Privacy issues
  - Data breaches: unlawful disclosure or access to sensitive data [GDPR16]
  - Service provider can get subpoenaed to reveal encryption key [Gre14]

Outsourcing Data to the Cloud

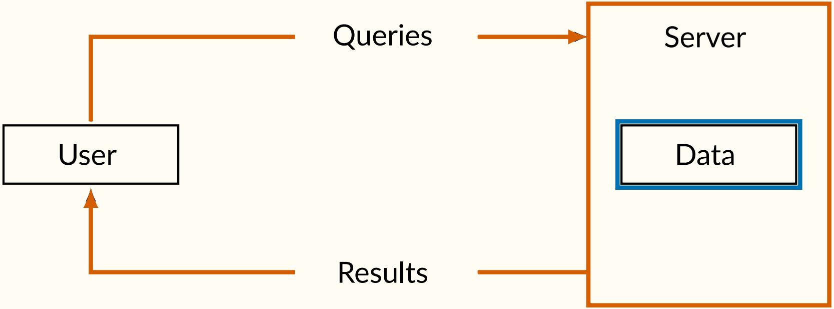


Outsourcing Data to the Cloud



Snapshot attack

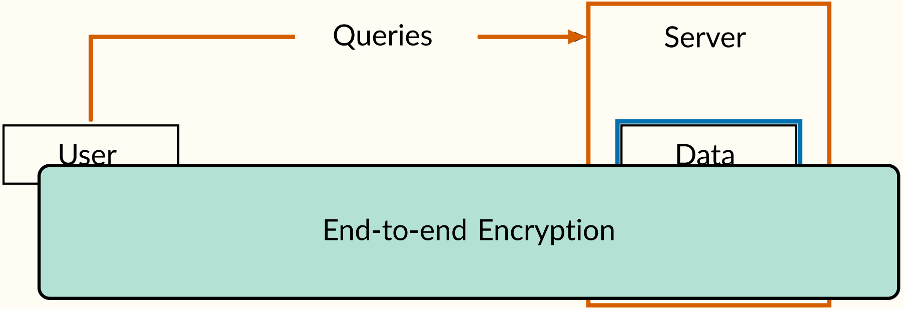
Outsourcing Data to the Cloud



Persistent attack

Snapshot attack

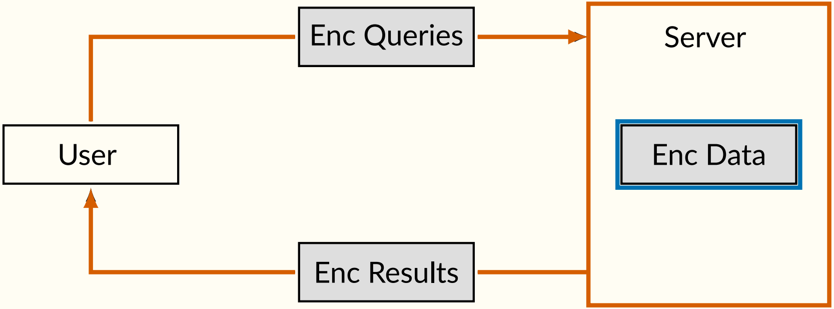
Outsourcing Data to the Cloud



Persistent attack

Snapshot attack

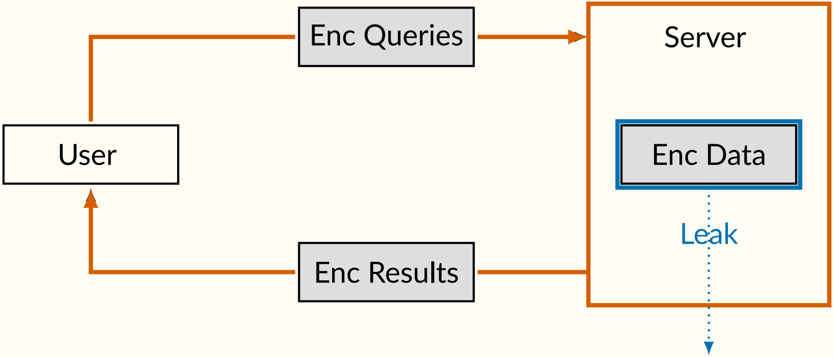
Outsourcing Data to the Cloud



Persistent attack

Snapshot attack

Outsourcing Data to the Cloud



Enc Queries

Server

User

Enc Data

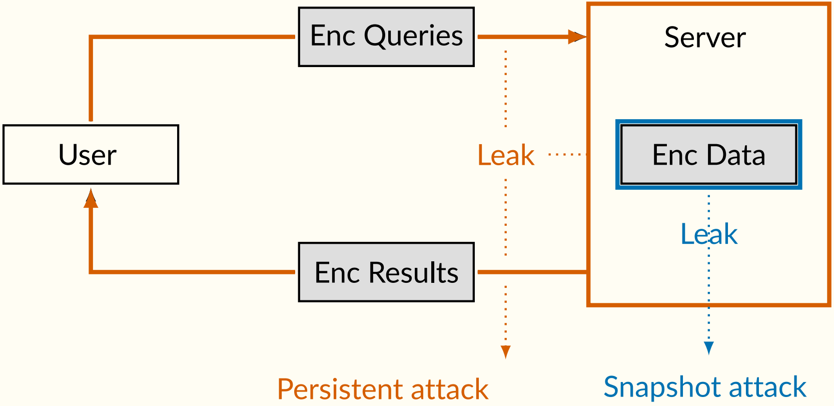
Leak

Enc Results

Persistent attack

Snapshot attack

Outsourcing Data to the Cloud



Outsourcing Data to the Cloud

New Schemes and System

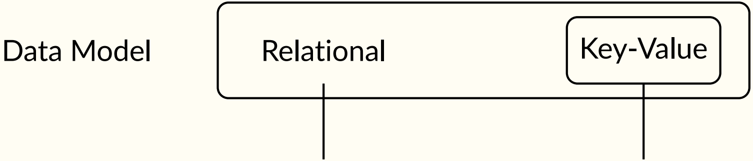
- Efficiency
- Small Leakage
- Expressivity: SQL / Relational Model
- Query Optimization
- Legacy Compatibility

Background

Background

Data Model	Relational	Key-Value

Background



Data Model

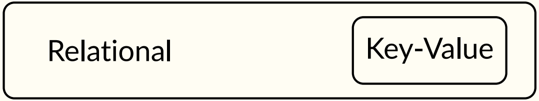
Relational

Key-Value

- SQL: Turing Complete
- Linear in DB length

- Boolean Algebra
- Linear in Result length

Background



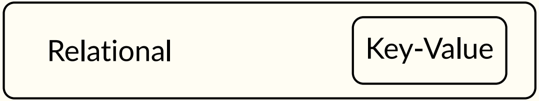
Relational

Key-Value

Trusted HW

- SCPA: TrustedDB [BS11]
- FPGA: Cipherbase [ABE+13]
- SGX: StealthDB [GVG19], Bunker [AKM19]
- Memory / Workload Limit
- Intrusive SW changes

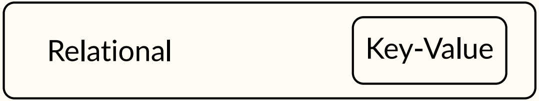
Background



Generic

- FHE [Gen09]: hides result length
- ORAM [GO96]: hides access pattern
- Smallest leakage but inefficient

Background

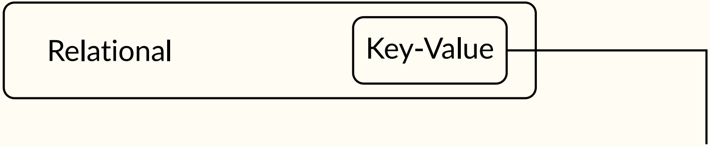


Relational

Key-Value

Trade leakage for higher efficiency

Background



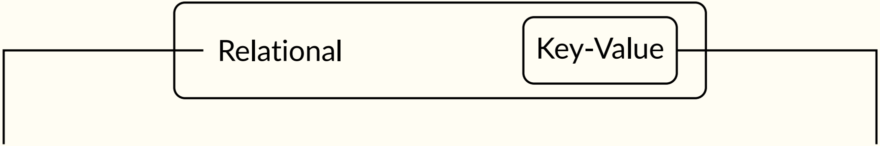
Relational

Key-Value

SSE [SWP00,CGKO06]

- Optimal in KV
- Small Leakage
- Systems
  - BlindSeer [PKV+14]
  - ESPADA [CJJ+14]

Background



Relational

Key-Value

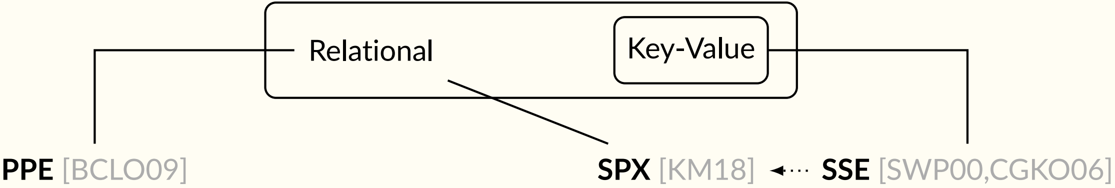
PPE [BCLO09]

- Optimal
- Large Leakage
- Optimization
- Legacy
- Systems
  - CryptDB [PRNB11]
  - Monomi [TKMZ13]

SSE [SWP00,CGKO06]

- Optimal in KV
- Small Leakage
- Systems
  - BlindSeer [PKV+14]
  - ESPADA [CJJ+14]

Background

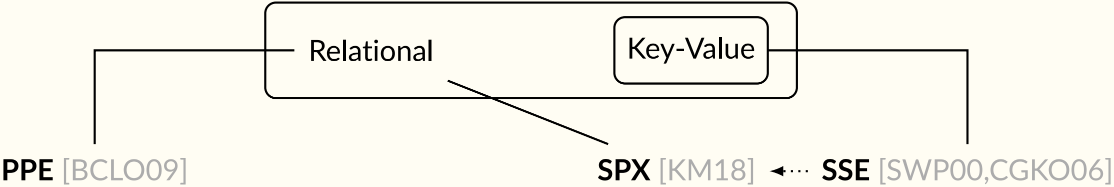


- Optimal
- Large Leakage
- Optimization
- Legacy
- Systems
  - CryptDB [PRNB11]
  - Monomi [TKMZ13]

- Quadratic
- Small Leakage
- Lack Opt.
- Lack Legacy
- Lack System

- Optimal in KV
- Small Leakage
- Systems
  - BlindSeer [PKV+14]
  - ESPADA [CJJ+14]

Background


Relational

Key-Value

PPE [BCLO09]

SPX [KM18] ◂⋯ **SSE** [SWP00,CGKO06]

PPE [BCLO09]

- Optimal
- Large Leakage
- Optimization
- Legacy
- Systems
  - CryptDB [PRNB11]
  - Monomi [TKMZ13]

SPX [KM18]

- Quadratic
- Small Leakage
- Lack Opt.
- Lack Legacy
- Lack System

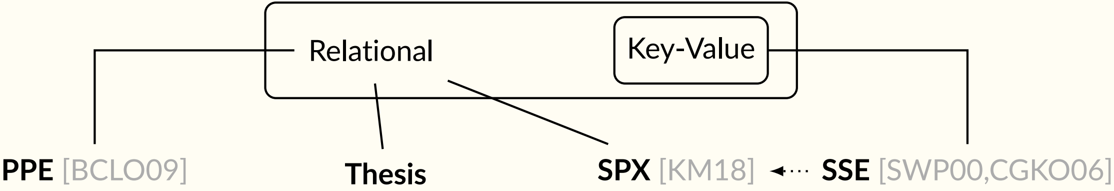
TPC-H showed

- 1000× query overhead
- 252× space overhead

SSE [SWP00,CGKO06]

- Optimal in KV
- Small Leakage
- Systems
  - BlindSeer [PKV+14]
  - ESPADA J+14]

Background



- Optimal
- Large Leakage
- Optimization
- Legacy
- Systems
  - CryptDB [PRNB11]
  - Monomi [TKMZ13]

- Optimal
- Smaller Leakage
- Std. & New Optimization
- Legacy
- System

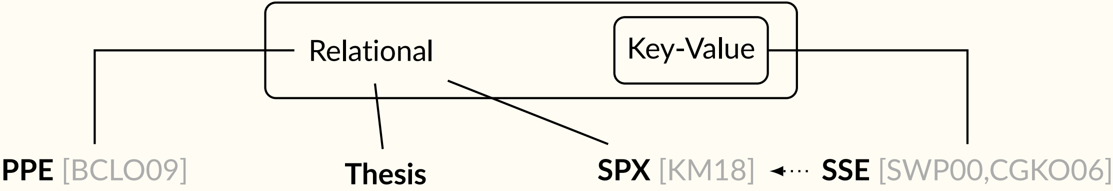
- Quadratic
- Small Leakage
- Lack Opt.
- Lack Legacy
- Lack System

TPC-H showed

- 1000× query overhead
- 252× space overhead

- Optimal in KV
- Small Leakage
- Systems
  - BlindSeer [PKV+14]
  - ESPADA J+14]

Background



Relational

Key-Value

PPE [BCLO09]

Thesis

SPX [KM18] ◄···· **SSE** [SWP00,CGKO06]

- Optimal
- Large Leakage
- Optimization
- Legacy
- Systems
  - CryptDB [PRNB11]

4× overhead

- Optimal
- Smaller Leakage
- Std. & New Optimization
- Legacy
- System

4× overhead

- Quadratic
- Small Leakage
- Lack Opt.
- Lack Legacy
- Lack System

TPC-H showed

- 1000× query overhead
- 252× space overhead

- Optimal in KV
- Small Leakage
- Systems
  - BlindSeer [PKV+14]
  - ESPADA J+14]

Relational Data

Relational Data

Tables and Operators: Select σ, Project π, Join $\bowtie$ (Conjunctive Queries).

Relational Data

Tables and Operators: Select σ, Project π, Join $\bowtie$ (Conjunctive Queries).

Customer

Name	Pay	Nation
Alice	VISA	US
Bob	PayPal	US
Bob	VISA	MEX

Supplier

Name	Nation
Intel	US
IBM	US
Intel	MEX
Arca	MEX

Relational Data

Tables and Operators: Select σ, Project π, Join $\bowtie$ (Conjunctive Queries).

Customer

Name	Pay	Nation
Alice	VISA	US
Bob	PayPal	US
Bob	VISA	MEX

Supplier

Name	Nation
Intel	US
IBM	US
Intel	MEX
Arca	MEX

Select Customer.Name, Supplier.Name
From Customer **Join** Supplier **On** Nation
Where Pay = VISA **And** Nation = US

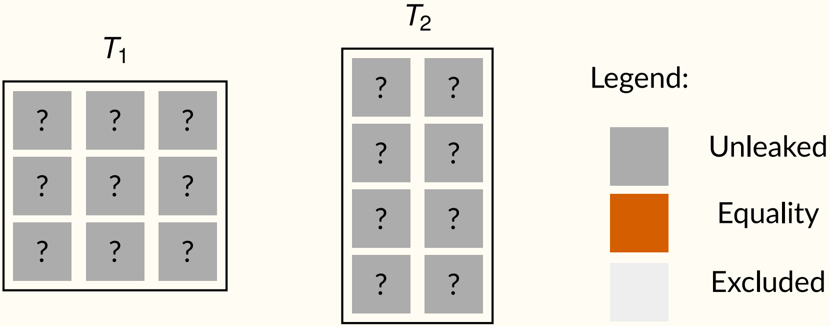
Security

Security of Encrypted Relational Database

Informally, Server learns nothing about data/query beyond the **defined leakage** [CGKO06].

Security of Encrypted Relational Database

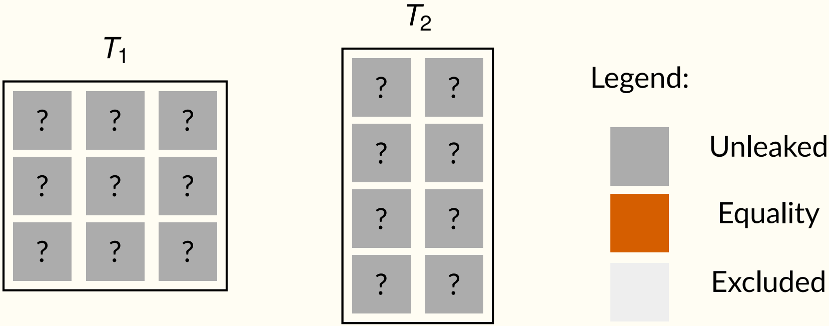
Informally, Server learns nothing about data/query beyond the **defined leakage** [CGKO06].



- Setup leaks: table dimensions, # tables

Security of Encrypted Relational Database

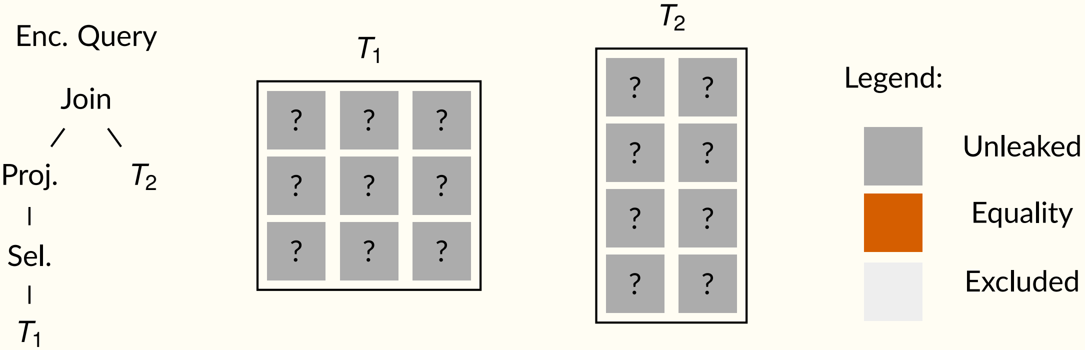
Informally, Server learns nothing about data/query beyond the **defined leakage** [CGKO06].



- Setup leaks: table dimensions, # tables
- Query leaks: **equality pattern**, result size, access, repetition.

Security of Encrypted Relational Database

Informally, Server learns nothing about data/query beyond the **defined leakage** [CGKO06].



T_1

?	?	?
?	?	?
?	?	?

T_2

?	?
?	?
?	?
?	?

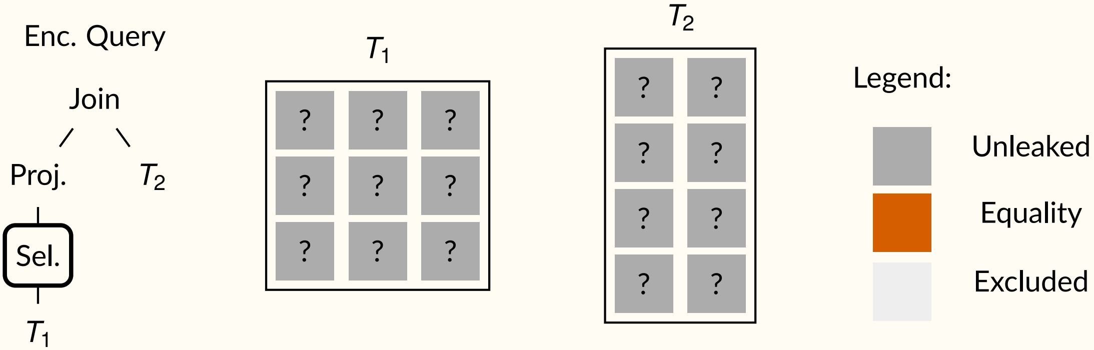
Legend:

▨	Unleaked
🟧	Equality
□	Excluded

- Setup leaks: table dimensions, # tables
- Query leaks: **equality pattern**, result size, access, repetition.

Security of Encrypted Relational Database

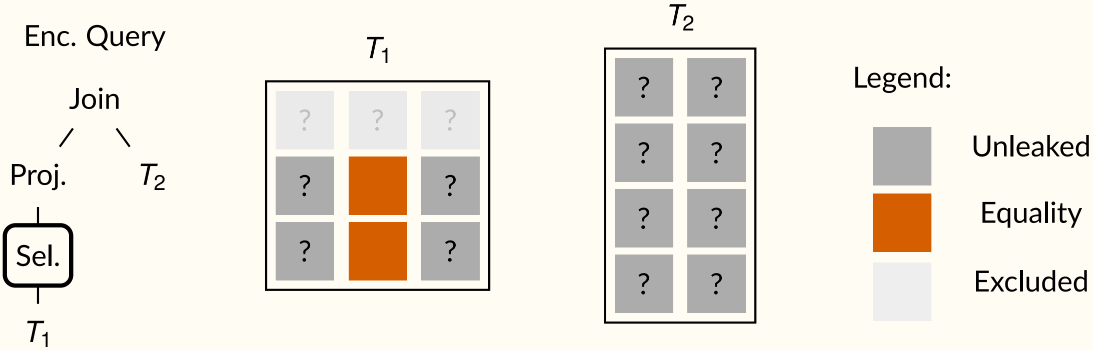
Informally, Server learns nothing about data/query beyond the **defined leakage** [CGKO06].



- Setup leaks: table dimensions, # tables
- Query leaks: **equality pattern**, result size, access, repetition.

Security of Encrypted Relational Database

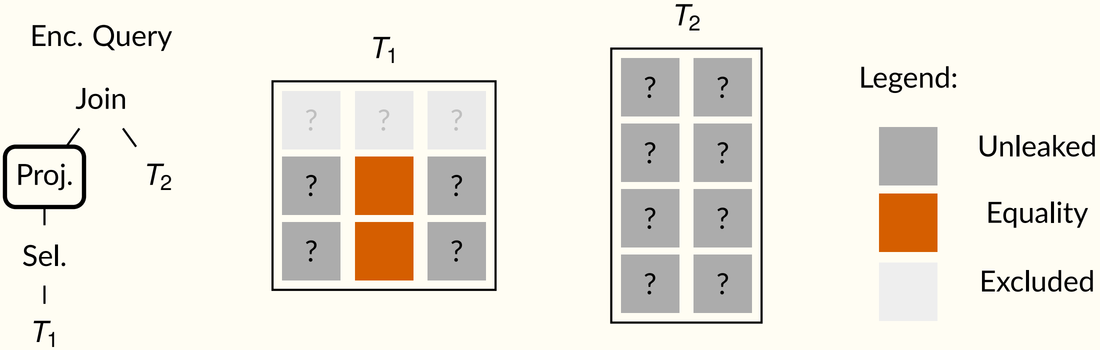
Informally, Server learns nothing about data/query beyond the **defined leakage** [CGKO06].



- Setup leaks: table dimensions, # tables
- Query leaks: **equality pattern**, result size, access, repetition.

Security of Encrypted Relational Database

Informally, Server learns nothing about data/query beyond the **defined leakage** [CGKO06].



T_1

?	?	?
?	🟧	?
?	🟧	?

T_2

?	?
?	?
?	?
?	?

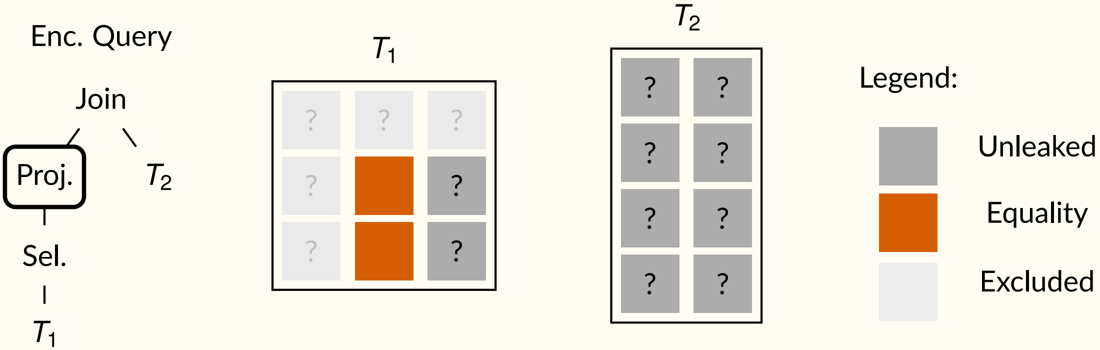
Legend:

▨	Unleaked
🟧	Equality
⬜	Excluded

- Setup leaks: table dimensions, # tables
- Query leaks: **equality pattern**, result size, access, repetition.

Security of Encrypted Relational Database

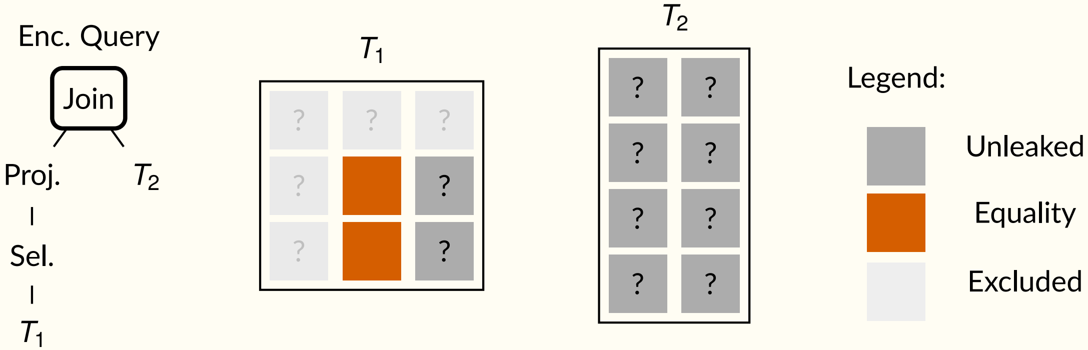
Informally, Server learns nothing about data/query beyond the **defined leakage** [CGKO06].



- Setup leaks: table dimensions, # tables
- Query leaks: **equality pattern**, result size, access, repetition.

Security of Encrypted Relational Database

Informally, Server learns nothing about data/query beyond the **defined leakage** [CGKO06].



Enc. Query

Join

Proj.

T_2

Sel.

T_1

T_1

?	?	?
?	🟧	?
?	🟧	?

T_2

?	?
?	?
?	?
?	?

Legend:

⬛ Unleaked

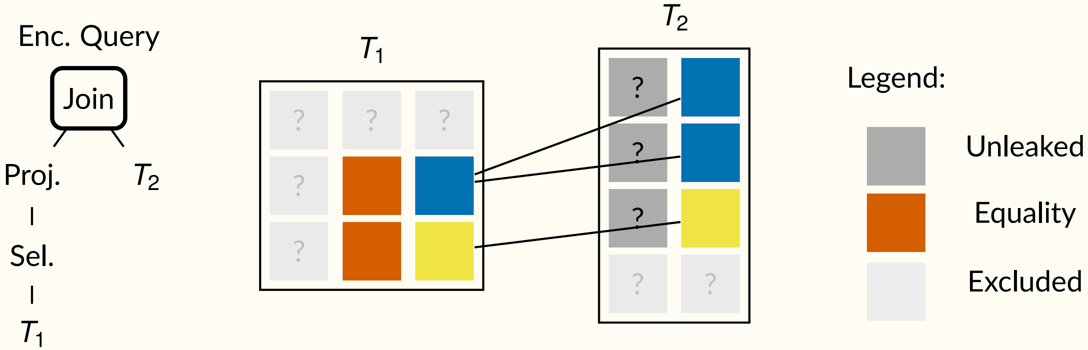
🟧 Equality

⬜ Excluded

- Setup leaks: table dimensions, # tables
- Query leaks: **equality pattern**, result size, access, repetition.

Security of Encrypted Relational Database

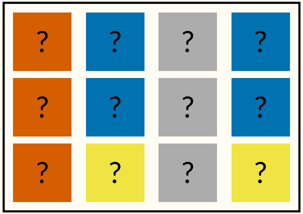
Informally, Server learns nothing about data/query beyond the **defined leakage** [CGKO06].



- Setup leaks: table dimensions, # tables
- Query leaks: **equality pattern**, result size, access, repetition.

Security of Encrypted Relational Database

Informally, Server learns nothing about data/query beyond the **defined leakage** [CGKO06].



🟧 ?	🟦 ?	⬜ ?	🟦 ?
🟧 ?	🟦 ?	⬜ ?	🟦 ?
🟧 ?	🟨 ?	⬜ ?	🟨 ?

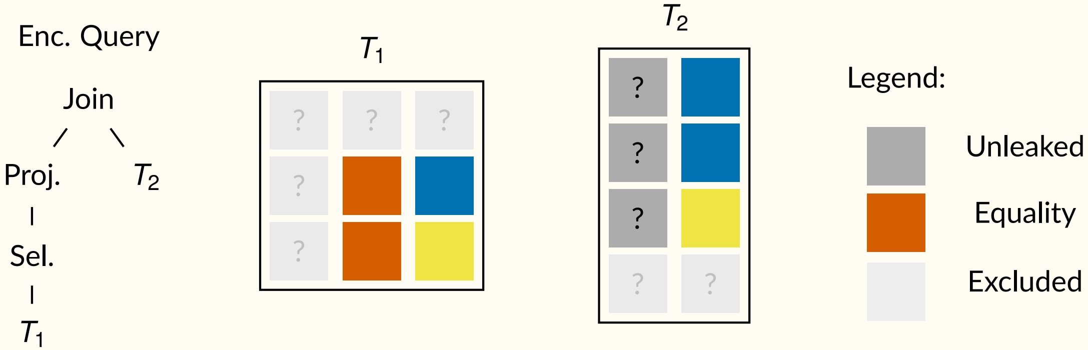
Legend:

⬜	Unleaked
🟧	Equality
▫	Excluded

- Setup leaks: table dimensions, # tables
- Query leaks: **equality pattern**, result size, access, repetition.

Security of Encrypted Relational Database

Informally, Server learns nothing about data/query beyond the **defined leakage** [CGKO06].



- Setup leaks: table dimensions, # tables
- Query leaks: **equality pattern**, result size, access, repetition.

Existing Approaches

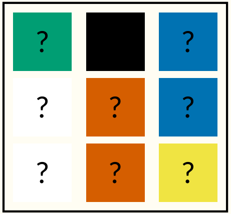
PPE-based schemes[1]

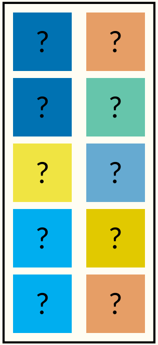
- Deterministic Encryption [BBO07]: ciphertexts reveal equality pattern
- Order-preserving Encryption [AKSX04]: ciphertexts reveal ordering pattern

[1] Quantization [HILM02], CryptDB [PRZB11], Monomi [TKMZ13]

PPE-based schemes[1]

- Deterministic Encryption [BBO07]: ciphertexts reveal equality pattern
- Order-preserving Encryption [AKSX04]: ciphertexts reveal ordering pattern

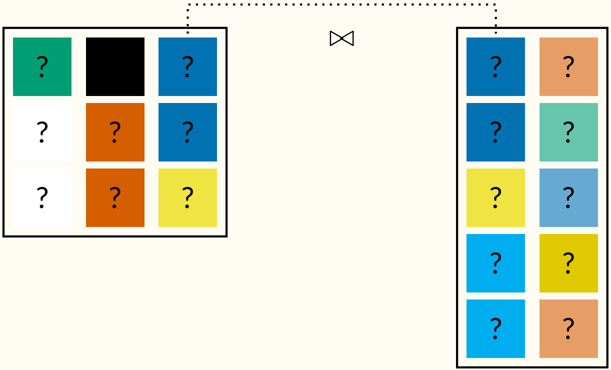




[1] Quantization [HILM02], CryptDB [PRZB11], Monomi [TKMZ13]

PPE-based schemes[1]

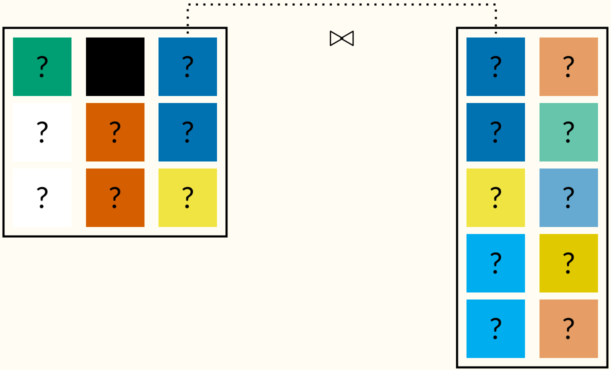
- Deterministic Encryption [BBO07]: ciphertexts reveal equality pattern
- Order-preserving Encryption [AKSX04]: ciphertexts reveal ordering pattern



[1] Quantization [HILM02], CryptDB [PRZB11], Monomi [TKMZ13]

PPE-based schemes[1]

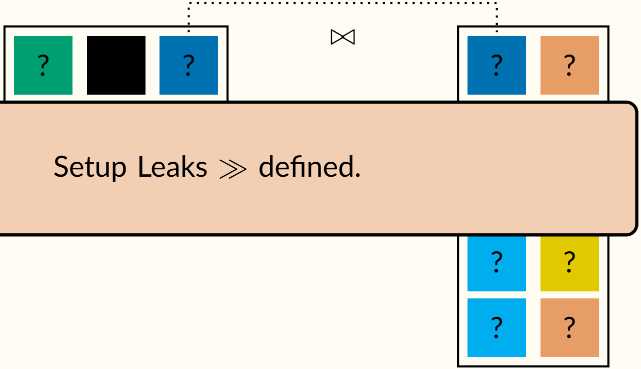
- Deterministic Encryption [BBO07]: ciphertexts reveal equality pattern
- Order-preserving Encryption [AKSX04]: ciphertexts reveal ordering pattern
- Data-recovery attacks [NKW15,DDC16,GSBNR17] for PPE-based schemes.



[1] Quantization [HILM02], CryptDB [PRZB11], Monomi [TKMZ13]

PPE-based schemes[1]

- Deterministic Encryption [BBO07]: ciphertexts reveal equality pattern
- Order-preserving Encryption [AKSX04]: ciphertexts reveal ordering pattern
- Data-recovery attacks [NKW15,DDC16,GSBNR17] for PPE-based schemes.



Setup Leaks $\gg$ defined.

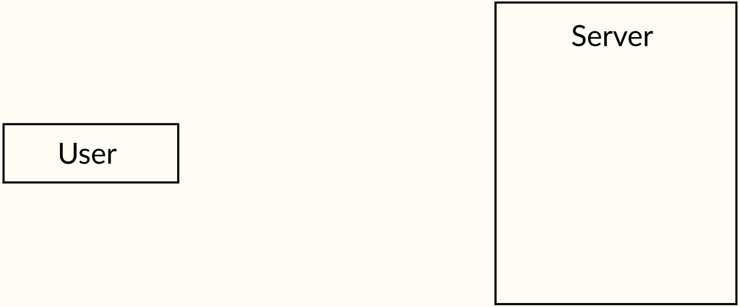
[1] Quantization [HILM02], CryptDB [PRZB11], Monomi [TKMZ13]

Structured Encryption (STE) [CK10]

- Generalization of index-based searchable encryption SSE [CGKO06].

Structured Encryption (STE) [CK10]

- Generalization of index-based searchable encryption SSE [CGKO06].

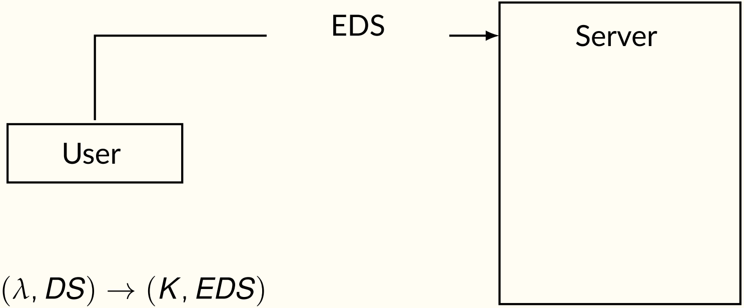


User

Server

Structured Encryption (STE) [CK10]

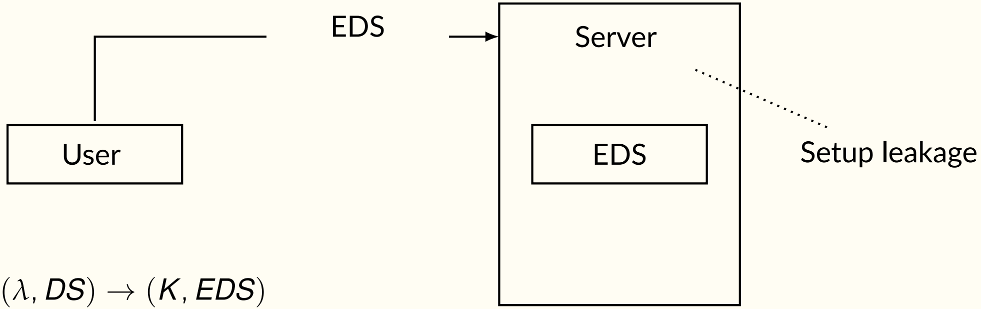
- Generalization of index-based searchable encryption SSE [CGKO06].



$Setup(\lambda, DS) \rightarrow (K, EDS)$

Structured Encryption (STE) [CK10]

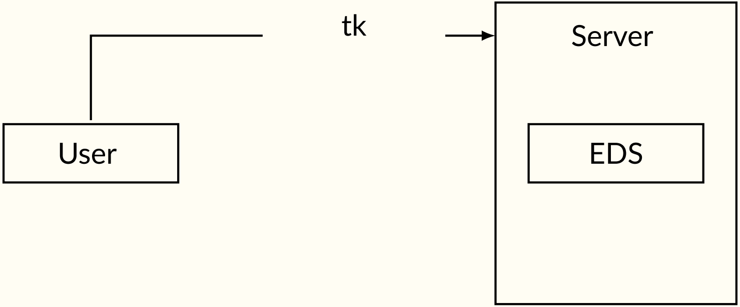
- Generalization of index-based searchable encryption SSE [CGKO06].



$Setup(\lambda, DS) \rightarrow (K, EDS)$

Structured Encryption (STE) [CK10]

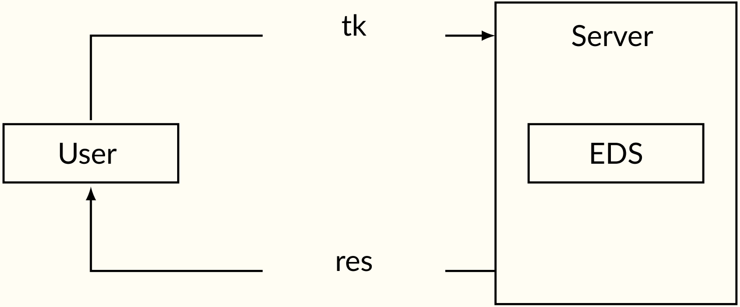
- Generalization of index-based searchable encryption SSE [CGKO06].



$$Token(K, query) \rightarrow tk$$

Structured Encryption (STE) [CK10]

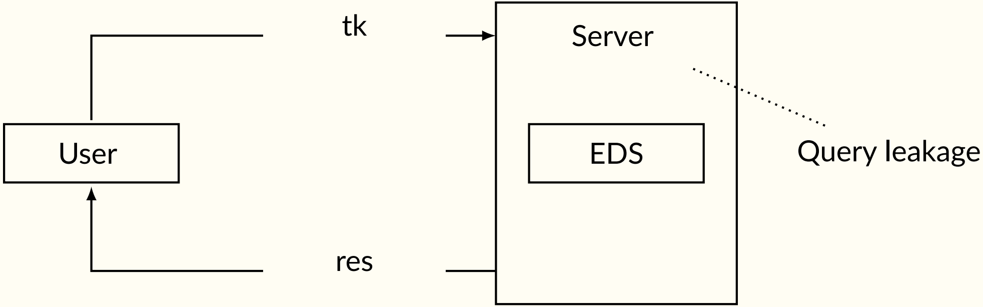
- Generalization of index-based searchable encryption SSE [CGKO06].



$Token(K, query) \rightarrow tk$ $Query(EDS, tk) \rightarrow res$

Structured Encryption (STE) [CK10]

- Generalization of index-based searchable encryption SSE [CGKO06].



$Token(K, query) \rightarrow tk$

$Query(EDS, tk) \rightarrow res$

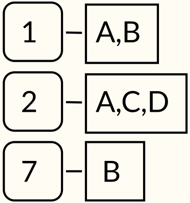
Encrypted Multimaps[2]

- Multimap associates a label with a tuple of values.

[2] [CGKO06],[CK10],[KPR12],[KP13],[CJJ+14],[Bost16],[BMO17],[AKM19]

Encrypted Multimaps[2]

- Multimap associates a label with a tuple of values.

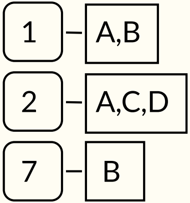


1	A,B
2	A,C,D
7	B

[2] [CGKO06],[CK10],[KPR12],[KP13],[CJJ+14],[Bost16],[BMO17],[AKM19]

Encrypted Multimaps[2]

- Multimap associates a label with a tuple of values.
-
    - $Get(2)$ returns (A, C, D).



1 – A,B

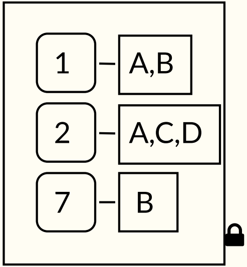
2 – A,C,D

7 – B

[2] [CGKO06],[CK10],[KPR12],[KP13],[CJJ+14],[Bost16],[BMO17],[AKM19]

Encrypted Multimaps[2]

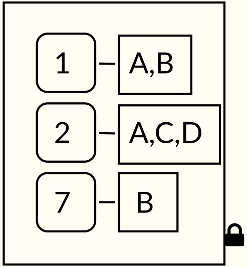
- Multimap associates a label with a tuple of values.
-
  - $\mathit{Get}(2)$ returns (A, C, D).
- STE schemes for multimap
  - $\mathit{Setup}(\lambda, \mathit{MM}) \rightarrow (K, \mathit{EMM})$
  - $\mathit{Token}(K, 2) \rightarrow \mathit{tk}$
  - $\mathit{Query}(\mathit{EMM}, \mathit{tk}) \rightarrow (A, C, D)$



[2] [CGKO06],[CK10],[KPR12],[KP13],[CJJ+14],[Bost16],[BMO17],[AKM19]

Encrypted Multimaps[2]

- Multimap associates a label with a tuple of values.
- - $Get(2)$ returns (A, C, D).
- STE schemes for multimap
  - $Setup(\lambda, MM) \rightarrow (K, EMM)$
  - $Token(K, 2) \rightarrow tk$
  - $Query(EMM, tk) \rightarrow (A, C, D)$
- E.g. Pibas[CJJ+14]: linear size. Setup leaks: total size. Query leaks: volume, query repetition, access.

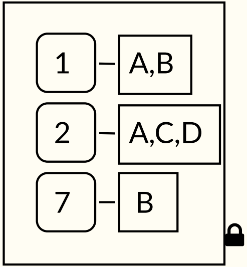


1	A,B
2	A,C,D
7	B

[2] [CGKO06],[CK10],[KPR12],[KP13],[CJJ+14],[Bost16],[BMO17],[AKM19]

Encrypted Multimaps[2]

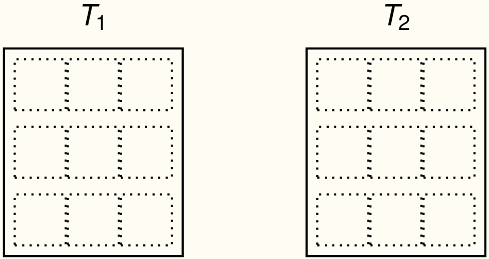
- Multimap associates a label with a tuple of values.
-
    - $Get(2)$ returns (A, C, D).
- STE schemes for multimap
    - $Setup(\lambda, MM) \rightarrow (K, EMM)$
    - $Token(K, 2) \rightarrow tk$
    - $Query(EMM, tk) \rightarrow (A, C, D)$
- E.g. Pibas[CJJ+14]: linear size. Setup leaks: total size.
  Query leaks: volume, query repetition, access.
- E.g. Zero-leakage constructions[KMO18,KM19].



[2] [CGKO06],[CK10],[KPR12],[KP13],[CJJ+14],[Bost16],[BMO17],[AKM19]

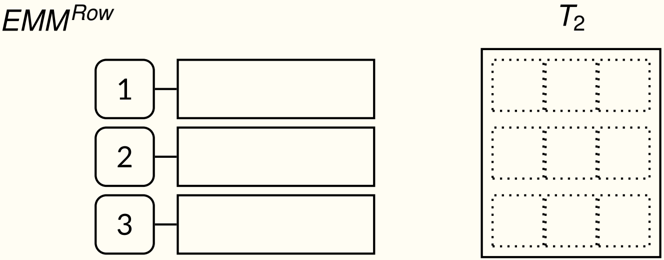
Encrypted Multimap-based Construction, SPX [KM18]

Represent tables and operators as multimaps. Example: Join



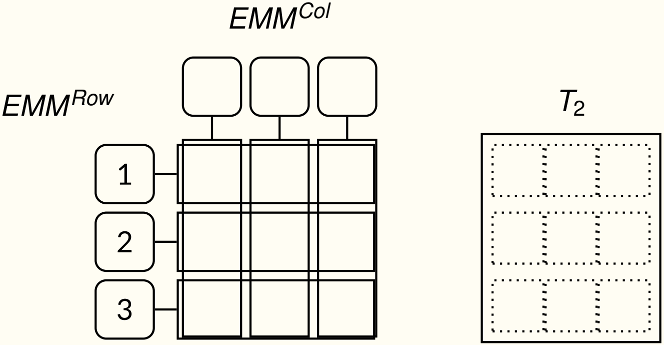
Encrypted Multimap-based Construction, SPX [KM18]

Represent tables and operators as multimaps. Example: Join



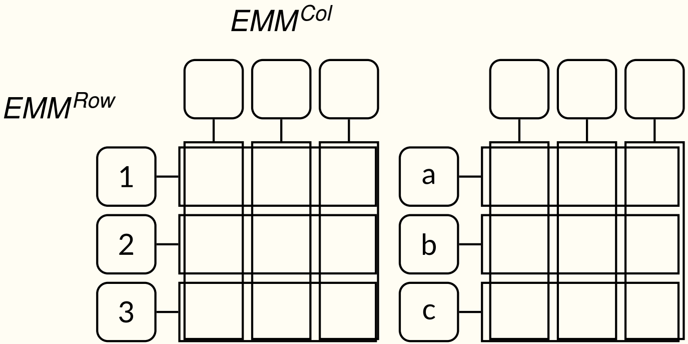
Encrypted Multimap-based Construction, SPX [KM18]

Represent tables and operators as multimaps. Example: Join



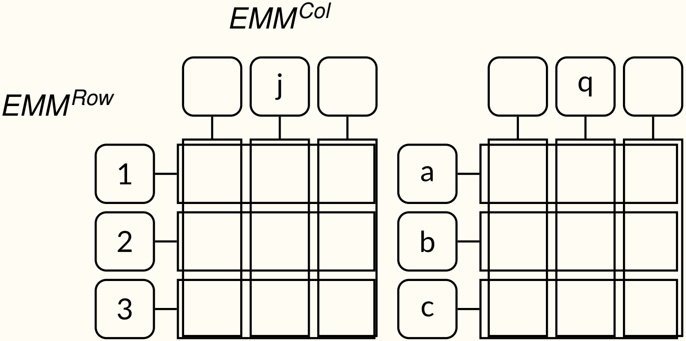
Encrypted Multimap-based Construction, SPX [KM18]

Represent tables and operators as multimaps. Example: Join



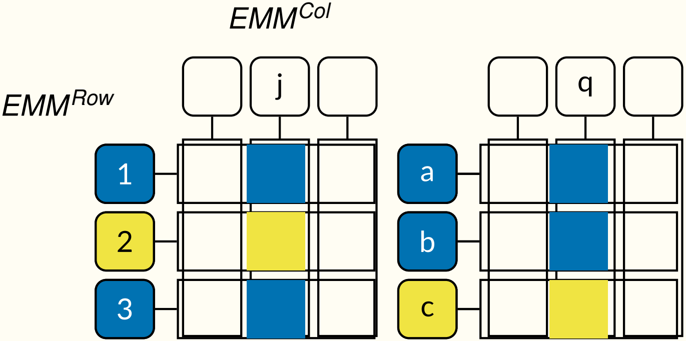
Encrypted Multimap-based Construction, SPX [KM18]

Represent tables and operators as multimaps. Example: Join



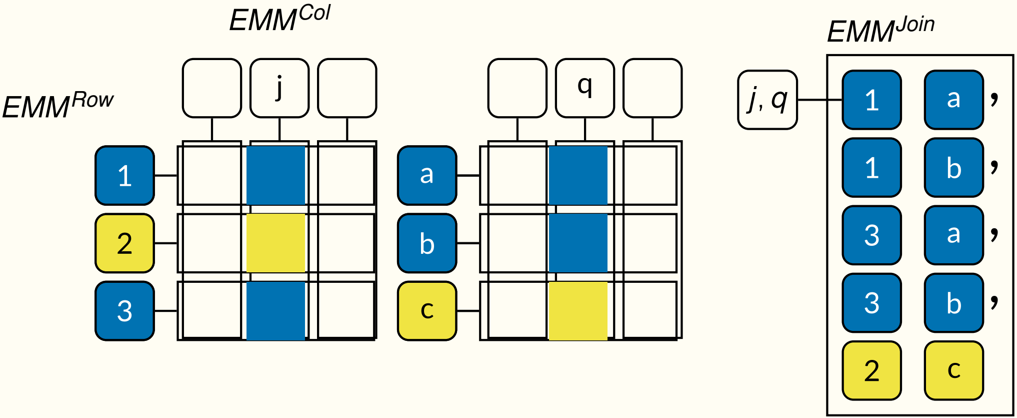
Encrypted Multimap-based Construction, SPX [KM18]

Represent tables and operators as multimaps. Example: Join



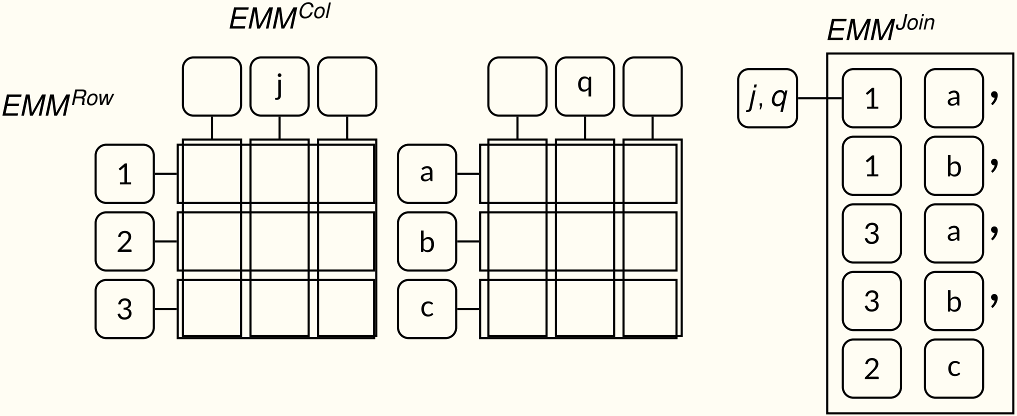
Encrypted Multimap-based Construction, SPX [KM18]

Represent tables and operators as multimaps. Example: Join



Encrypted Multimap-based Construction, SPX [KM18]

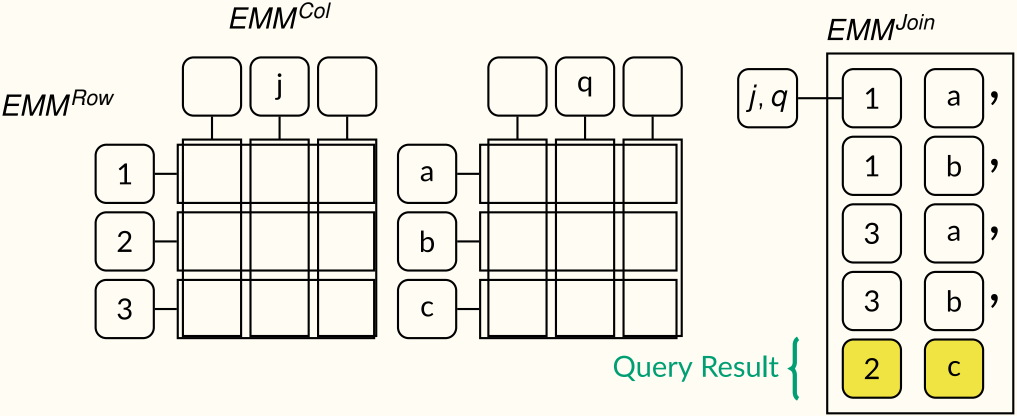
Represent tables and operators as multimaps. Example: Join



Encrypted as EMMs

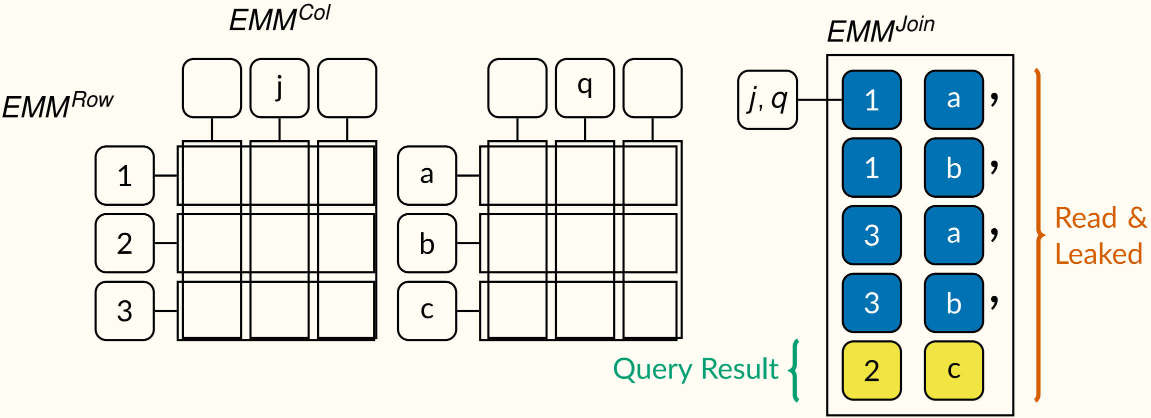
Encrypted Multimap-based Construction, SPX [KM18]

Represent tables and operators as multimaps. Example: Join



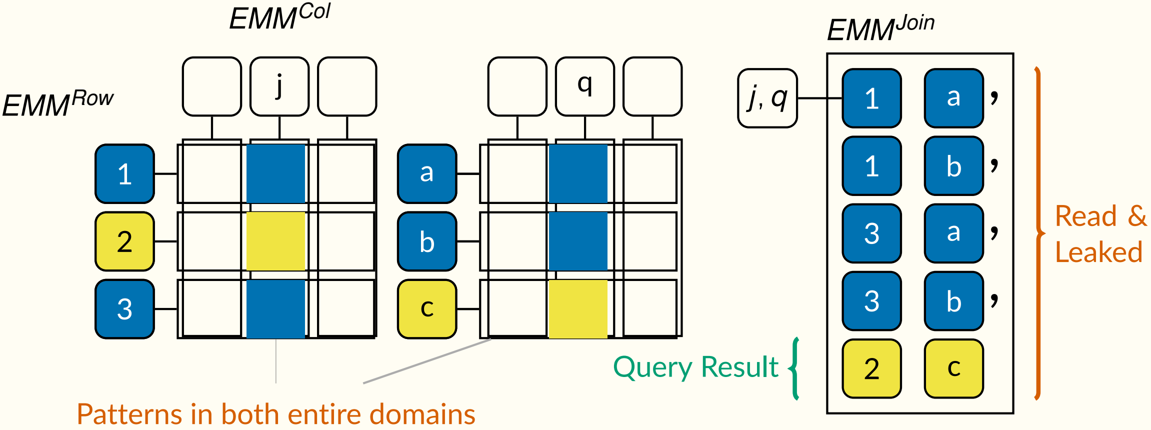
Encrypted Multimap-based Construction, SPX [KM18]

Represent tables and operators as multimaps. Example: Join



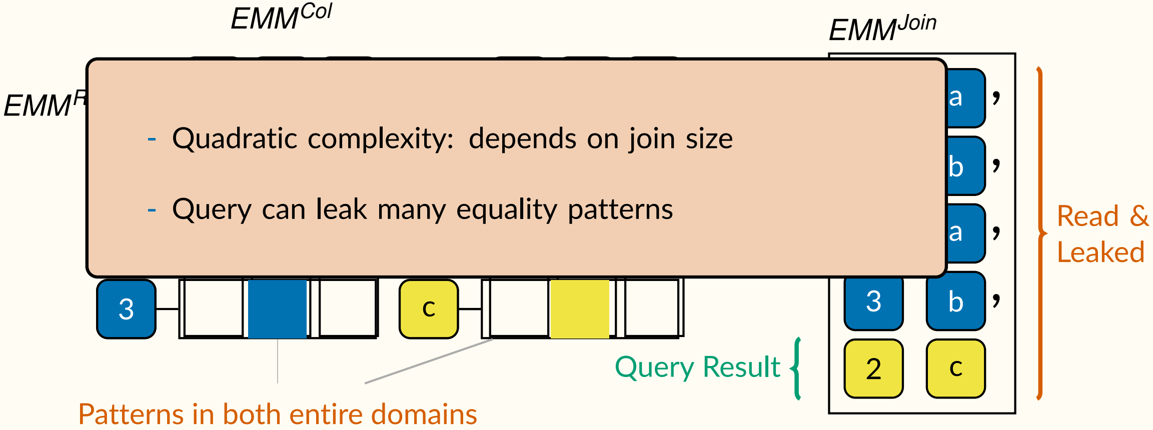
Encrypted Multimap-based Construction, SPX [KM18]

Represent tables and operators as multimaps. Example: Join



Encrypted Multimap-based Construction, SPX [KM18]

Represent tables and operators as multimaps. Example: Join



- Quadratic complexity: depends on join size
- Query can leak many equality patterns

Thesis

Outline[3]

- OPX: encrypted query optimization, quadratic
- PKFK: (1) optimal scheme in relational setting; (2) improved locality; (3) reduced leakage
- KafeDB: legacy compatible encrypted SQL database

[3] Joint work with Tarik Moataz at Aroki Systems, Seny Kamara and Stan Zdonik at Brown University. Works include OPX [KMZZ20], KafeDB [ZKMZ21], PKFK [In submission].

Outline[3]

- OPX: encrypted query optimization, quadratic
- **PKFK: (1) optimal scheme in relational setting; (2) improved locality;** (3) reduced leakage
- KafeDB: legacy compatible encrypted SQL database

[3] Joint work with Tarik Moataz at Aroki Systems, Seny Kamara and Stan Zdonik at Brown University. Works include OPX [KMZZ20], KafeDB [ZKMZ21], PKFK [In submission].

Outline[3]

- OPX: encrypted query optimization, quadratic
- **PKFK: (1) optimal scheme in relational setting; (2) improved locality;** (3) reduced leakage
- **KafeDB: legacy compatible encrypted SQL database**

[3] Joint work with Tarik Moataz at Aroki Systems, Seny Kamara and Stan Zdonik at Brown University. Works include OPX [KMZZ20], KafeDB [ZKMZ21], PKFK [In submission].

Overview of PKFK Scheme

- Optimal encrypted Join: matches plaintext Join in complexity [BTAO15]

Overview of PKFK Scheme

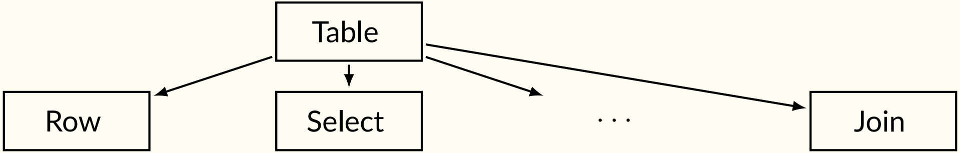
- Optimal encrypted Join: matches plaintext Join in complexity [BTAO15]

- Improved leakage

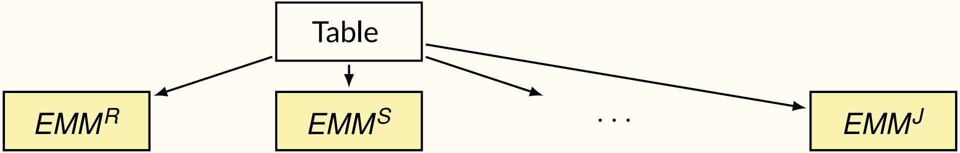
Overview of PKFK Scheme

- Optimal encrypted Join: matches plaintext Join in complexity [BTAO15]
- Improved leakage
- General techniques for STE:
- (1) Emulation: make STE legacy friendly
- (2) Colocation: increase STE locality

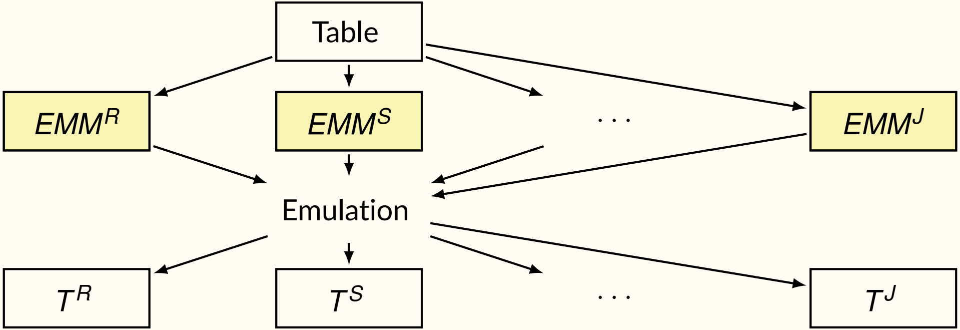
Overview of PKFK Scheme



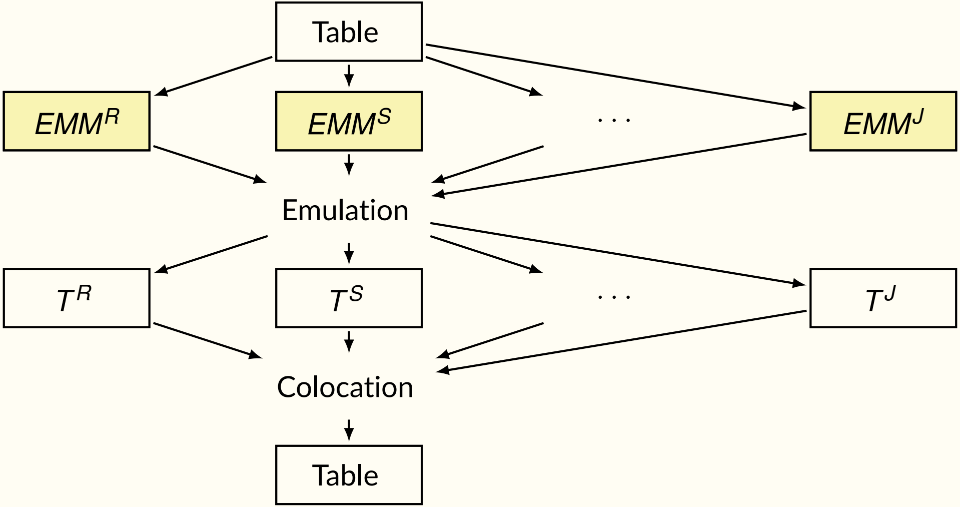
Overview of PKFK Scheme



Overview of PKFK Scheme



Overview of PKFK Scheme



Table

EMM^R

EMM^S

. . .

EMM^J

Emulation

T^R

T^S

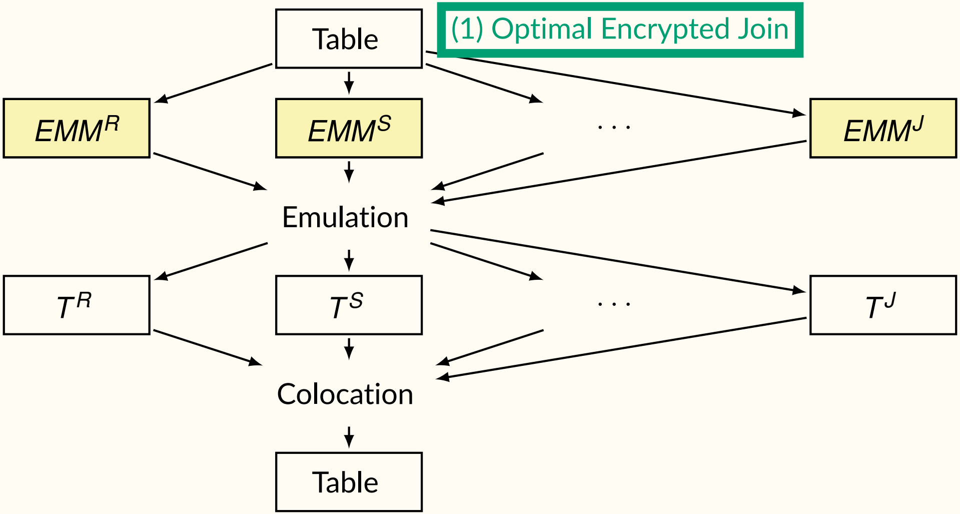
. . .

T^J

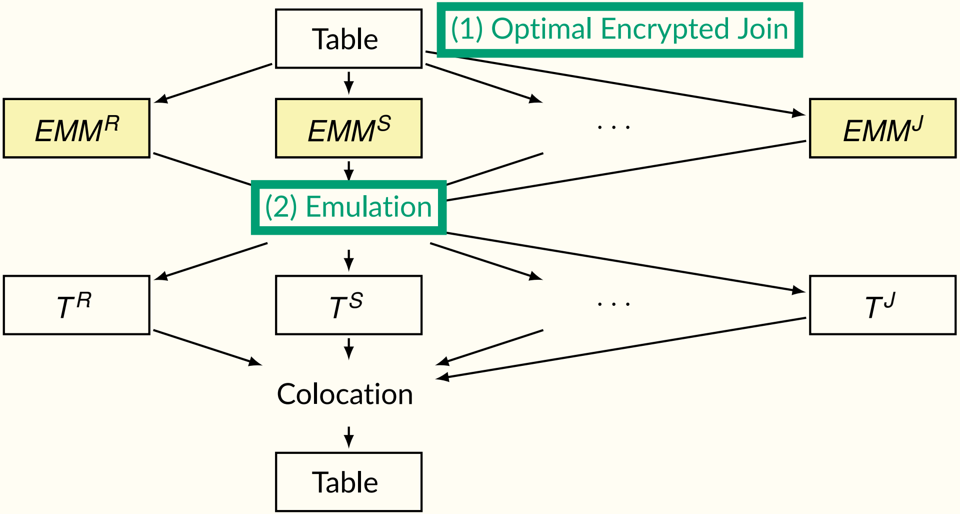
Colocation

Table

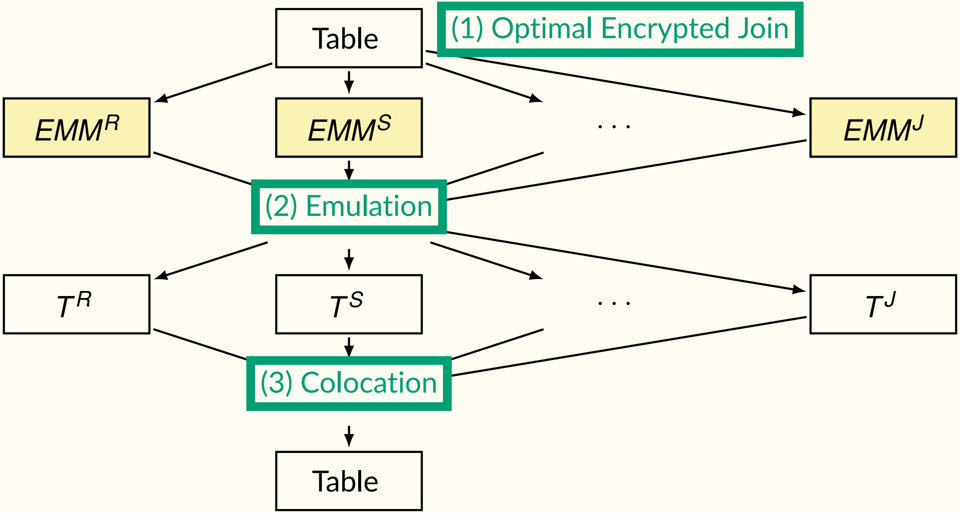
Overview of PKFK Scheme



Overview of PKFK Scheme



Overview of PKFK Scheme



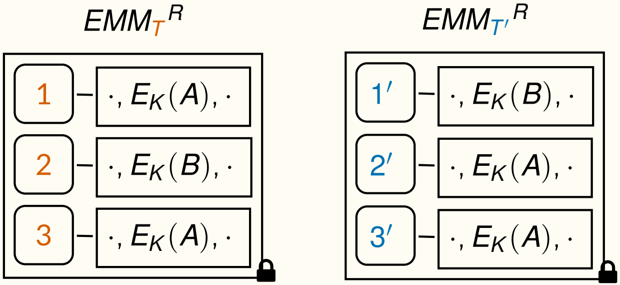
Optimal Encrypted Join

Optimal Encrypted Join

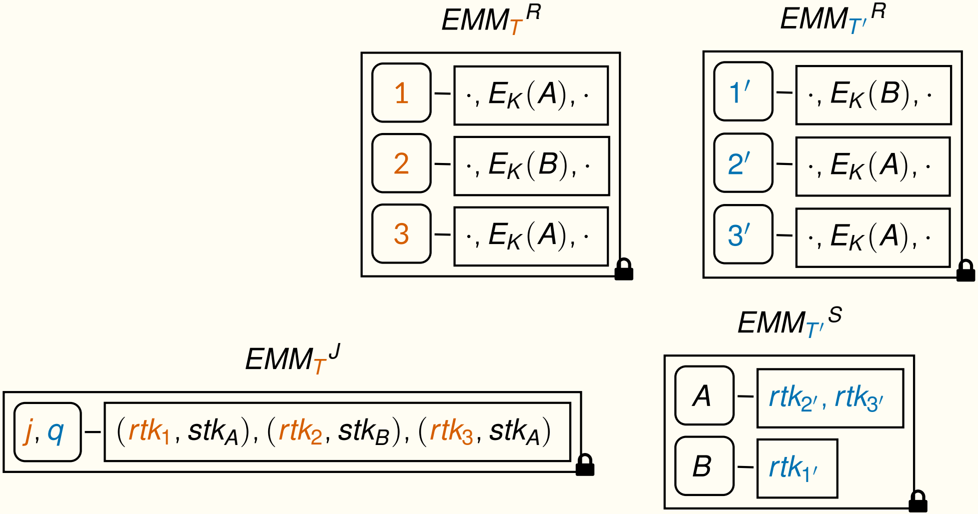
T		Col j	
1		A	
2		B	
3		A	

T'		Col q	
$1'$		B	
$2'$		A	
$3'$		A	

Optimal Encrypted Join



Optimal Encrypted Join



$EMM_T{}^R$

1	$\cdot, E_K(A), \cdot$
2	$\cdot, E_K(B), \cdot$
3	$\cdot, E_K(A), \cdot$

$EMM_{T'}{}^R$

$1'$	$\cdot, E_K(B), \cdot$
$2'$	$\cdot, E_K(A), \cdot$
$3'$	$\cdot, E_K(A), \cdot$

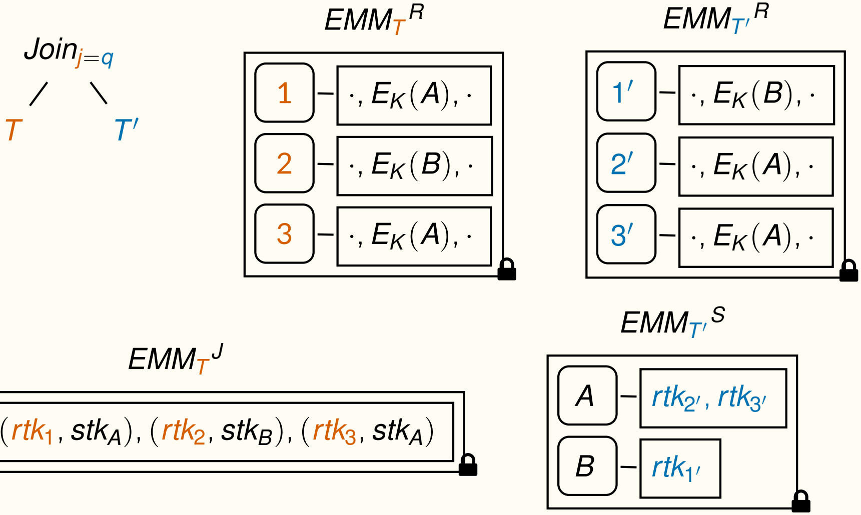
$EMM_T{}^J$

j, q	$(rtk_1, stk_A), (rtk_2, stk_B), (rtk_3, stk_A)$

$EMM_{T'}{}^S$

A	$rtk_{2'}, rtk_{3'}$
B	$rtk_{1'}$

Optimal Encrypted Join



$\mathit{Join}_{j=q}$

T $\quad$ T'

$EMM_T{}^R$

1	$\cdot, E_K(A), \cdot$
2	$\cdot, E_K(B), \cdot$
3	$\cdot, E_K(A), \cdot$

$EMM_{T'}{}^R$

$1'$	$\cdot, E_K(B), \cdot$
$2'$	$\cdot, E_K(A), \cdot$
$3'$	$\cdot, E_K(A), \cdot$

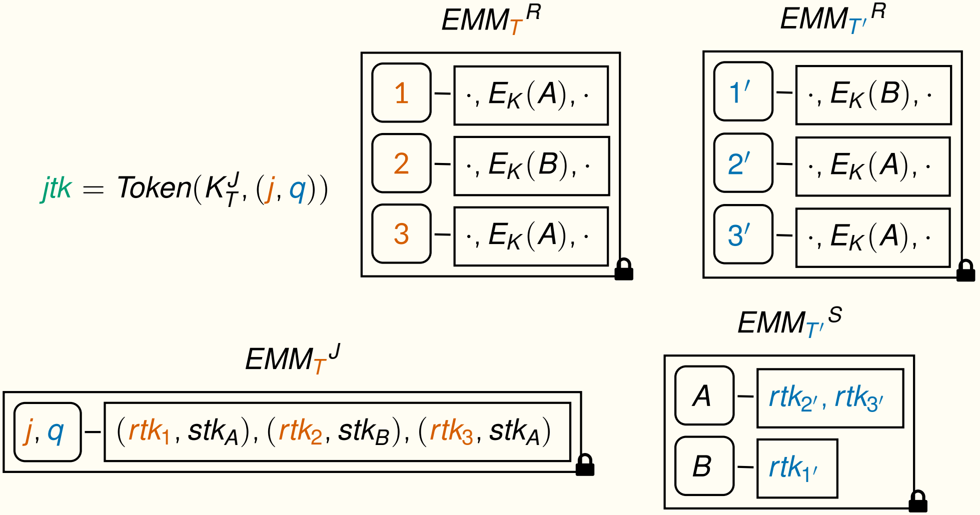
$EMM_T{}^J$

j, q	$(rtk_1, stk_A), (rtk_2, stk_B), (rtk_3, stk_A)$

$EMM_{T'}{}^S$

A	$rtk_{2'}, rtk_{3'}$
B	$rtk_{1'}$

Optimal Encrypted Join



$jtk = Token(K_T^J, (j, q))$

$EMM_T{}^R$

1 — $\cdot, E_K(A), \cdot$

2 — $\cdot, E_K(B), \cdot$

3 — $\cdot, E_K(A), \cdot$

$EMM_{T'}{}^R$

$1'$ — $\cdot, E_K(B), \cdot$

$2'$ — $\cdot, E_K(A), \cdot$

$3'$ — $\cdot, E_K(A), \cdot$

$EMM_T{}^J$

j, q — $(rtk_1, stk_A), (rtk_2, stk_B), (rtk_3, stk_A)$

$EMM_{T'}{}^S$

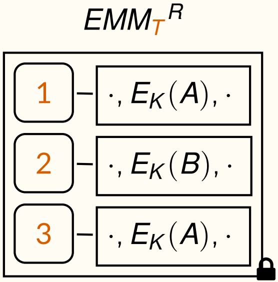
A — $rtk_{2'}, rtk_{3'}$

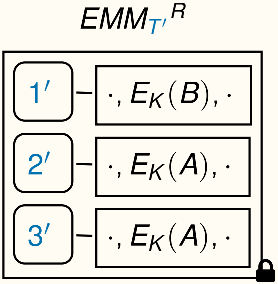
B — $rtk_{1'}$

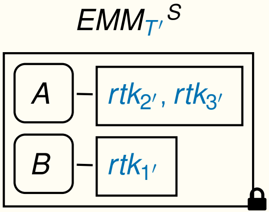
Optimal Encrypted Join

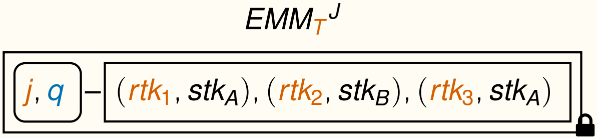
$Query(EDB, jtk)$

```
For $(rtk, stk)$ in $Query(EMM_T{}^J, jtk)$
    $r \leftarrow Query(EMM_T{}^R, rtk)$
    For $rtk'$ in $Query(EMM_{T'}{}^S, stk)$
        $R' \leftarrow R' \cup Query(EMM_{T'}{}^R, rtk')$
    Output $(r, R')$
```





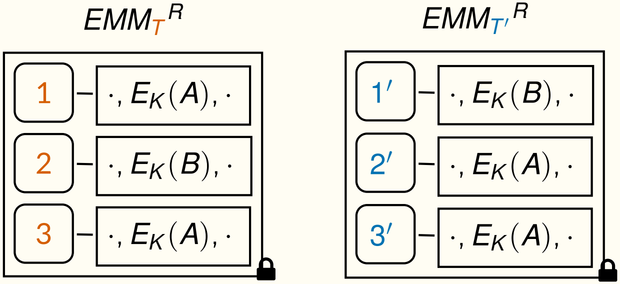




Optimal Encrypted Join

$Query(EDB, jtk)$

```
For $(rtk, stk)$ in $Query(EMM_T{}^J, jtk)$
    $r \leftarrow Query(EMM_T{}^R, rtk)$
    For $rtk'$ in $Query(EMM_{T'}{}^S, stk)$
        $R' \leftarrow R' \cup Query(EMM_{T'}{}^R, rtk')$
    Output $(r, R')$
```

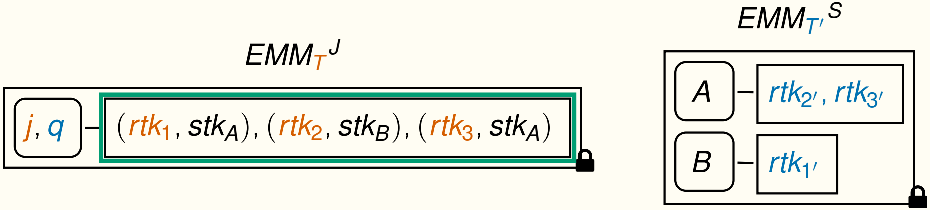


$EMM_T{}^R$

1	$\cdot, E_K(A), \cdot$
2	$\cdot, E_K(B), \cdot$
3	$\cdot, E_K(A), \cdot$

$EMM_{T'}{}^R$

$1'$	$\cdot, E_K(B), \cdot$
$2'$	$\cdot, E_K(A), \cdot$
$3'$	$\cdot, E_K(A), \cdot$



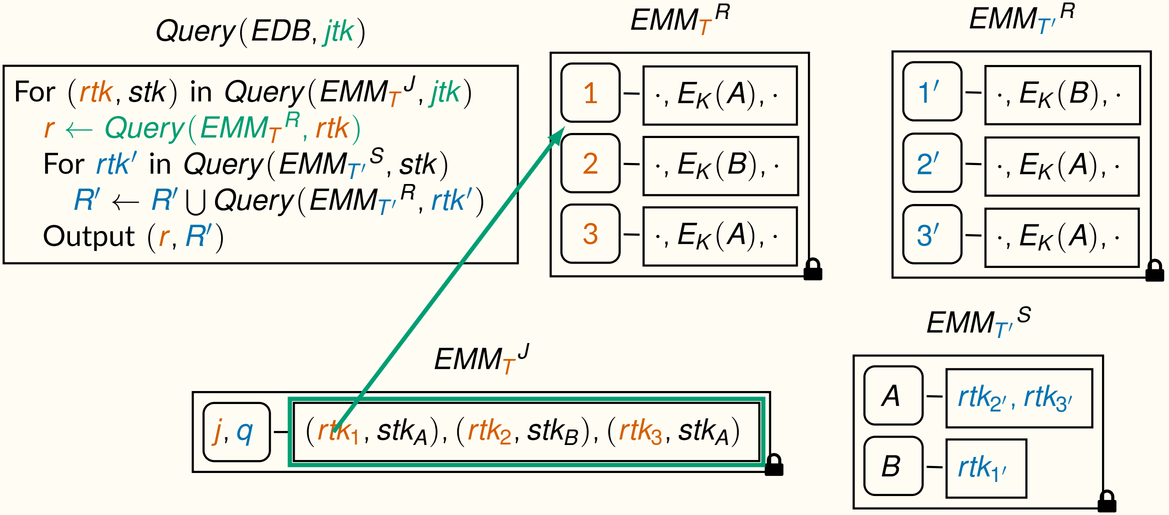
$EMM_T{}^J$

j, q	$(rtk_1, stk_A), (rtk_2, stk_B), (rtk_3, stk_A)$

$EMM_{T'}{}^S$

A	$rtk_{2'}, rtk_{3'}$
B	$rtk_{1'}$

Optimal Encrypted Join



$Query(EDB, jtk)$

```
For $(rtk, stk)$ in $Query(EMM_T{}^J, jtk)$
    $r \leftarrow Query(EMM_T{}^R, rtk)$
    For $rtk'$ in $Query(EMM_{T'}{}^S, stk)$
        $R' \leftarrow R' \cup Query(EMM_{T'}{}^R, rtk')$
    Output $(r, R')$
```

$EMM_T{}^R$

1 — $\cdot, E_K(A), \cdot$

2 — $\cdot, E_K(B), \cdot$

3 — $\cdot, E_K(A), \cdot$

$EMM_{T'}{}^R$

$1'$ — $\cdot, E_K(B), \cdot$

$2'$ — $\cdot, E_K(A), \cdot$

$3'$ — $\cdot, E_K(A), \cdot$

$EMM_T{}^J$

j, q — $(rtk_1, stk_A), (rtk_2, stk_B), (rtk_3, stk_A)$

$EMM_{T'}{}^S$

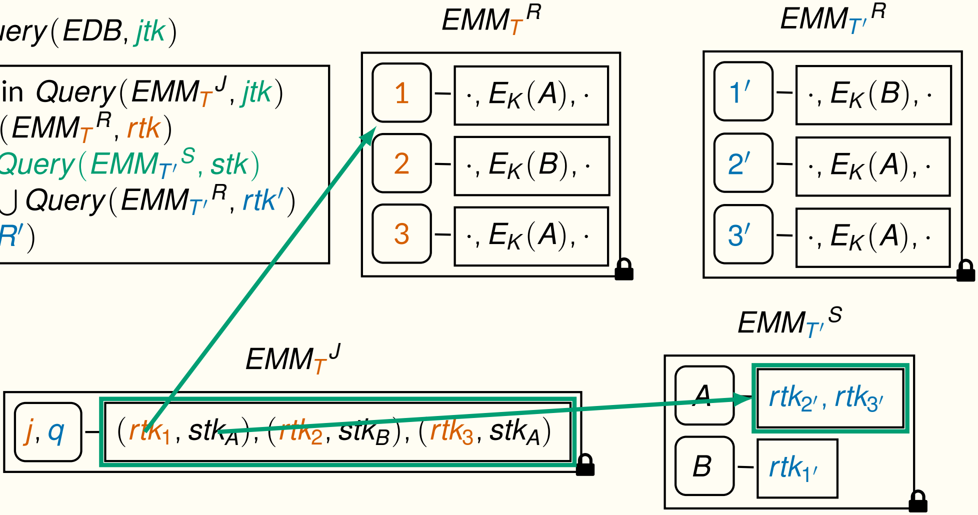
A — $rtk_{2'}, rtk_{3'}$

B — $rtk_{1'}$

Optimal Encrypted Join

$Query(EDB, jtk)$

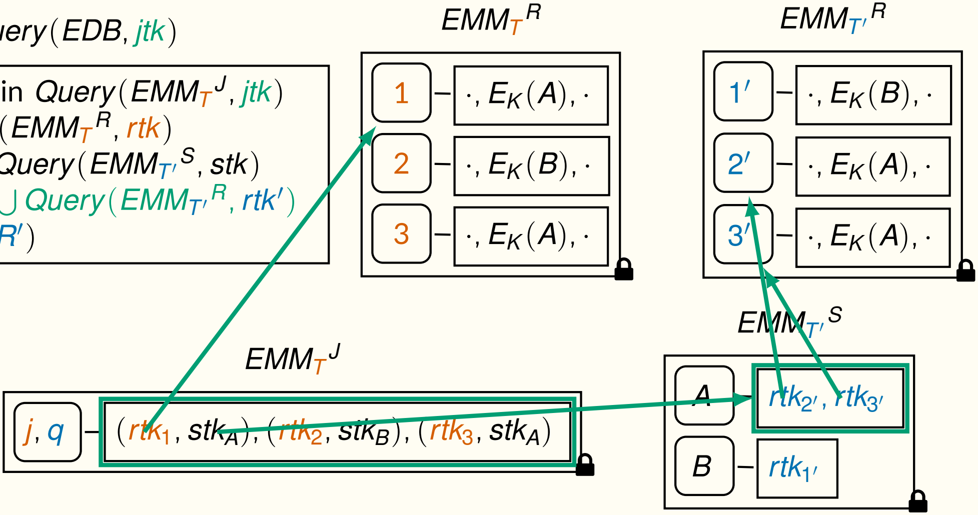
```
For $(rtk, stk)$ in $Query(EMM_T{}^J, jtk)$
    $r \leftarrow Query(EMM_T{}^R, rtk)$
    For $rtk'$ in $Query(EMM_{T'}{}^S, stk)$
        $R' \leftarrow R' \cup Query(EMM_{T'}{}^R, rtk')$
    Output $(r, R')$
```


$EMM_T{}^R$
1 — $\cdot, E_K(A), \cdot$
2 — $\cdot, E_K(B), \cdot$
3 — $\cdot, E_K(A), \cdot$
$EMM_{T'}{}^R$
$1'$ — $\cdot, E_K(B), \cdot$
$2'$ — $\cdot, E_K(A), \cdot$
$3'$ — $\cdot, E_K(A), \cdot$
$EMM_T{}^J$
j, q — $(rtk_1, stk_A), (rtk_2, stk_B), (rtk_3, stk_A)$
$EMM_{T'}{}^S$
A — $rtk_{2'}, rtk_{3'}$
B — $rtk_{1'}$

Optimal Encrypted Join

$Query(EDB, jtk)$

```
For $(rtk, stk)$ in $Query(EMM_T{}^J, jtk)$
    $r \leftarrow Query(EMM_T{}^R, rtk)$
    For $rtk'$ in $Query(EMM_{T'}{}^S, stk)$
        $R' \leftarrow R' \bigcup Query(EMM_{T'}{}^R, rtk')$
    Output $(r, R')$
```



$EMM_T{}^R$

1 — $\cdot, E_K(A), \cdot$

2 — $\cdot, E_K(B), \cdot$

3 — $\cdot, E_K(A), \cdot$

$EMM_{T'}{}^R$

$1'$ — $\cdot, E_K(B), \cdot$

$2'$ — $\cdot, E_K(A), \cdot$

$3'$ — $\cdot, E_K(A), \cdot$

$EMM_T{}^J$

j, q — $(rtk_1, stk_A), (rtk_2, stk_B), (rtk_3, stk_A)$

$EMM_{T'}{}^S$

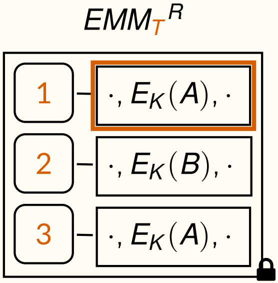
A — $rtk_{2'}, rtk_{3'}$

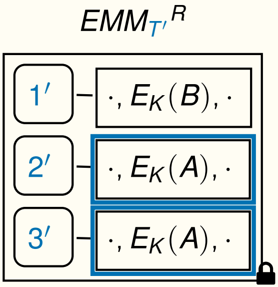
B — $rtk_{1'}$

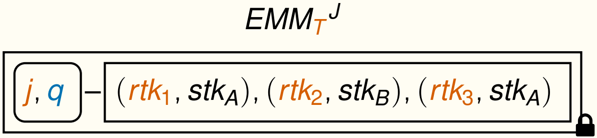
Optimal Encrypted Join

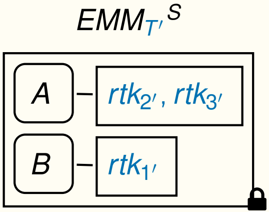
$Query(EDB, jtk)$

```
For $(rtk, stk)$ in $Query(EMM_T{}^J, jtk)$
    $r \leftarrow Query(EMM_T{}^R, rtk)$
    For $rtk'$ in $Query(EMM_{T'}{}^S, stk)$
        $R' \leftarrow R' \cup Query(EMM_{T'}{}^R, rtk')$
    Output $(r, R')$
```


$EMM_T{}^R$
1 — $\cdot, E_K(A), \cdot$
2 — $\cdot, E_K(B), \cdot$
3 — $\cdot, E_K(A), \cdot$


$EMM_{T'}{}^R$
$1'$ — $\cdot, E_K(B), \cdot$
$2'$ — $\cdot, E_K(A), \cdot$
$3'$ — $\cdot, E_K(A), \cdot$

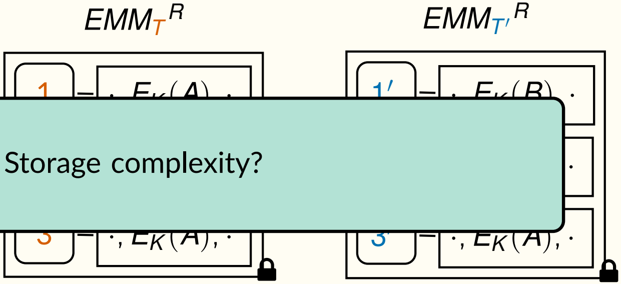




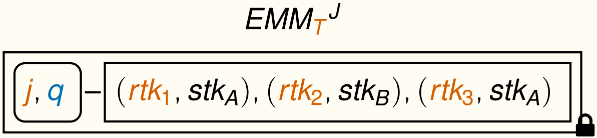
Optimal Encrypted Join

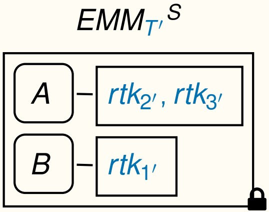
$Query(EDB, jtk)$

```
For $(rtk, stk)$ in $Query(EMM_T{}^J, jtk)$
    $r \leftarrow Query(EM$
    For $rtk'$ in $Quer$
        $R' \leftarrow R' \cup Qu$
    Output $(r, R')$
```



Storage complexity?



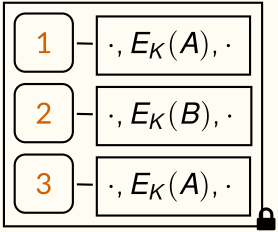


Optimal Encrypted Join

$Query(EDB, jtk)$

```
For $(rtk, stk)$ in $Query(EMM_T{}^J, jtk)$
    $r \leftarrow Query(EMM_T{}^R, rtk)$
    For $rtk'$ in $Query(EMM_{T'}{}^S, stk)$
        $R' \leftarrow R' \cup Query(EMM_{T'}{}^R, rtk')$
    Output $(r, R')$
```

$EMM_T{}^R$

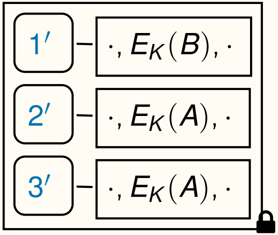


1 – $\cdot, E_K(A), \cdot$

2 – $\cdot, E_K(B), \cdot$

3 – $\cdot, E_K(A), \cdot$

$EMM_{T'}{}^R$

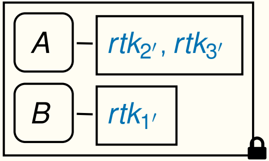


$1'$ – $\cdot, E_K(B), \cdot$

$2'$ – $\cdot, E_K(A), \cdot$

$3'$ – $\cdot, E_K(A), \cdot$

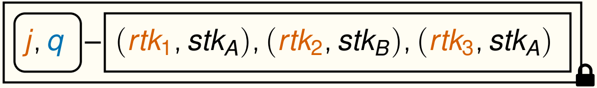
$EMM_{T'}{}^S$



A – $rtk_{2'}, rtk_{3'}$

B – $rtk_{1'}$

$EMM_T{}^J$

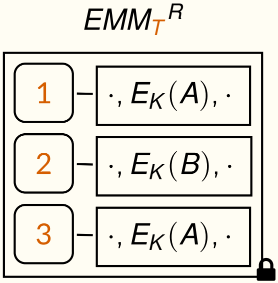


j, q – $(rtk_1, stk_A), (rtk_2, stk_B), (rtk_3, stk_A)$

Optimal Encrypted Join

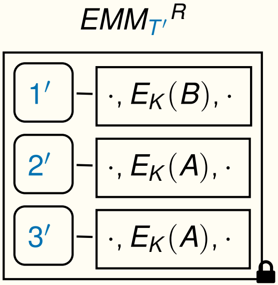
$Query(EDB, jtk)$

```
For $(rtk, stk)$ in $Query(EMM_T{}^J, jtk)$
    $r \leftarrow Query(EMM_T{}^R, rtk)$
    For $rtk'$ in $Query(EMM_{T'}{}^S, stk)$
        $R' \leftarrow R' \cup Query(EMM_{T'}{}^R, rtk')$
    Output $(r, R')$
```



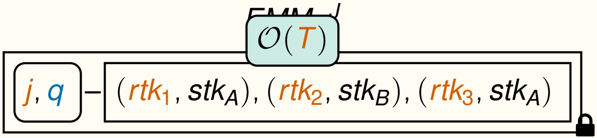
$EMM_T{}^R$

1	$\cdot, E_K(A), \cdot$
2	$\cdot, E_K(B), \cdot$
3	$\cdot, E_K(A), \cdot$



$EMM_{T'}{}^R$

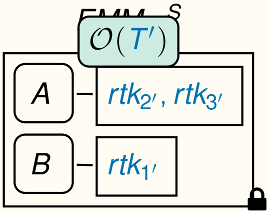
$1'$	$\cdot, E_K(B), \cdot$
$2'$	$\cdot, E_K(A), \cdot$
$3'$	$\cdot, E_K(A), \cdot$



EMM^J

$\mathcal{O}(T)$

j, q	$(rtk_1, stk_A), (rtk_2, stk_B), (rtk_3, stk_A)$



EMM^S

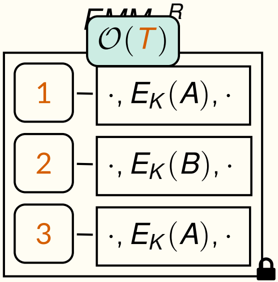
$\mathcal{O}(T')$

A	$rtk_{2'}, rtk_{3'}$
B	$rtk_{1'}$

Optimal Encrypted Join

$Query(EDB, jtk)$

```
For $(rtk, stk)$ in $Query(EMM_T{}^J, jtk)$
    $r \leftarrow Query(EMM_T{}^R, rtk)$
    For $rtk'$ in $Query(EMM_{T'}{}^S, stk)$
        $R' \leftarrow R' \cup Query(EMM_{T'}{}^R, rtk')$
    Output $(r, R')$
```



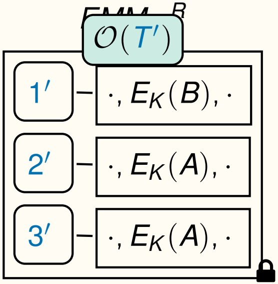
$EMM_T{}^R$

$\mathcal{O}(T)$

1 — $\cdot, E_K(A), \cdot$

2 — $\cdot, E_K(B), \cdot$

3 — $\cdot, E_K(A), \cdot$



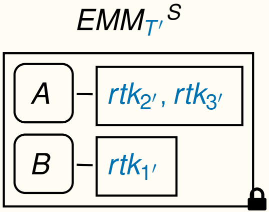
$EMM_{T'}{}^R$

$\mathcal{O}(T')$

$1'$ — $\cdot, E_K(B), \cdot$

$2'$ — $\cdot, E_K(A), \cdot$

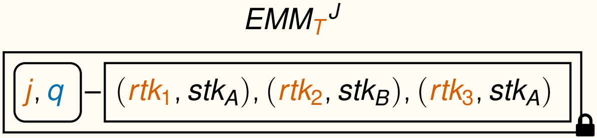
$3'$ — $\cdot, E_K(A), \cdot$



$EMM_{T'}{}^S$

A — $rtk_{2'}, rtk_{3'}$

B — $rtk_{1'}$



$EMM_T{}^J$

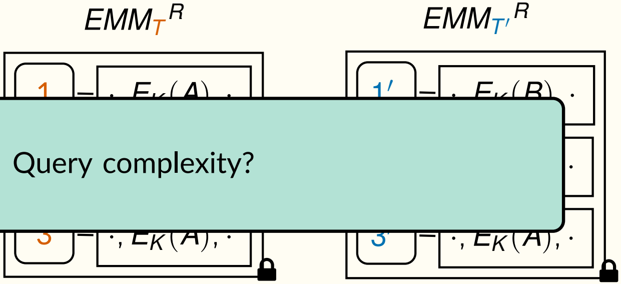
j, q — $(rtk_1, stk_A), (rtk_2, stk_B), (rtk_3, stk_A)$

Optimal Encrypted Join

$Query(EDB, jtk)$

```
For $(rtk, stk)$ in $Query(EMM_T{}^J, jtk)$
    $r \leftarrow Query(EM$
    For $rtk'$ in $Quer$
        $R' \leftarrow R' \bigcup Qu$
    Output $(r, R')$
```

Query complexity?



$EMM_T{}^R$

1 — $\cdot, E_K(A), \cdot$

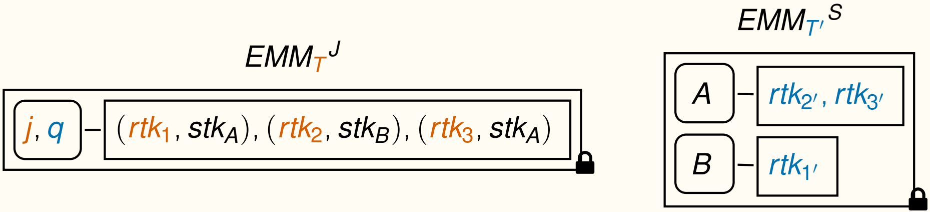
3 — $\cdot, E_K(A), \cdot$

$EMM_{T'}{}^R$

$1'$ — $\cdot, E_K(B), \cdot$

$\cdot$

$3'$ — $\cdot, E_K(A), \cdot$



$EMM_T{}^J$

j, q — $(rtk_1, stk_A), (rtk_2, stk_B), (rtk_3, stk_A)$

$EMM_{T'}{}^S$

A — $rtk_{2'}, rtk_{3'}$

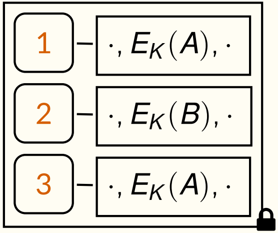
B — $rtk_{1'}$

Optimal Encrypted Join

$\mathit{Query}(\mathit{EDB}, \mathit{jtk})$

```
For $(rtk, stk)$ in $Query(EMM_T{}^J, jtk)$
    $r \leftarrow Query(EMM_T{}^R, rtk)$
    For $rtk'$ in $Query(EMM_{T'}{}^S, stk)$
        $R' \leftarrow R' \cup Query(EMM_{T'}{}^R, rtk')$
    Output $(r, R')$
```

$EMM_T{}^R$

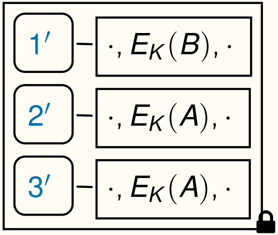


1 — $\cdot, E_K(A), \cdot$

2 — $\cdot, E_K(B), \cdot$

3 — $\cdot, E_K(A), \cdot$

$EMM_{T'}{}^R$

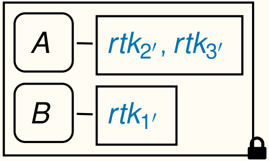


$1'$ — $\cdot, E_K(B), \cdot$

$2'$ — $\cdot, E_K(A), \cdot$

$3'$ — $\cdot, E_K(A), \cdot$

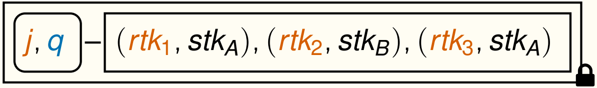
$EMM_{T'}{}^S$



A — $rtk_{2'}, rtk_{3'}$

B — $rtk_{1'}$

$EMM_T{}^J$



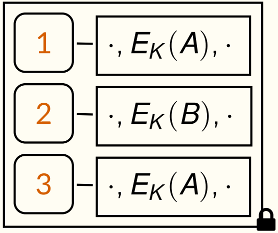
j, q — $(rtk_1, stk_A), (rtk_2, stk_B), (rtk_3, stk_A)$

Optimal Encrypted Join

$Query(EDB, jtk)$

```
For $(rtk, stk)$ in $Query(EMM_T{}^J, jtk)$
    $r \leftarrow Query(EMM_T{}^R, rtk)$
    For $rtk'$ in $Query(EMM_{T'}{}^S, stk)$
        $R' \leftarrow R' \cup Query(EMM_{T'}{}^R, rtk')$
    Output $(r, R')$
```

$EMM_T{}^R$

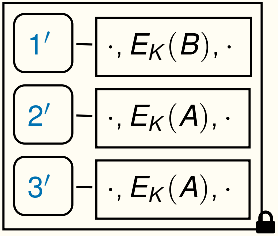


1 — $\cdot, E_K(A), \cdot$

2 — $\cdot, E_K(B), \cdot$

3 — $\cdot, E_K(A), \cdot$

$EMM_{T'}{}^R$

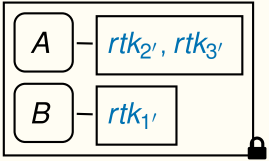


$1'$ — $\cdot, E_K(B), \cdot$

$2'$ — $\cdot, E_K(A), \cdot$

$3'$ — $\cdot, E_K(A), \cdot$

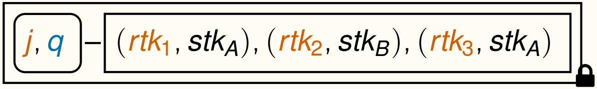
$EMM_{T'}{}^S$



A — $rtk_{2'}, rtk_{3'}$

B — $rtk_{1'}$

$EMM_T{}^J$

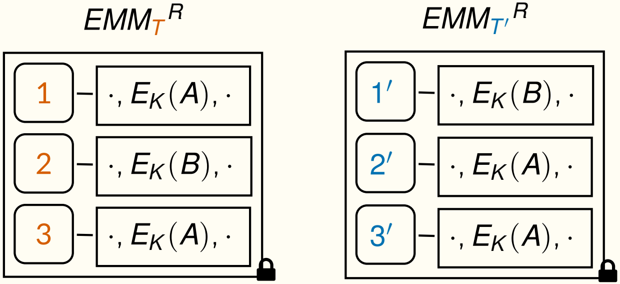


j, q — $(rtk_1, stk_A), (rtk_2, stk_B), (rtk_3, stk_A)$

Optimal Encrypted Join

$Query(EDB, jtk)$ $\mathcal{O}(T)$

```
For $(rtk, stk)$ in $Query(EMM_T{}^J, jtk)$
    $r \leftarrow Query(EMM_T{}^R, rtk)$
    For $rtk'$ in $Query(EMM_{T'}{}^S, stk)$
        $R' \leftarrow R' \cup Query(EMM_{T'}{}^R, rtk')$
    Output $(r, R')$
```

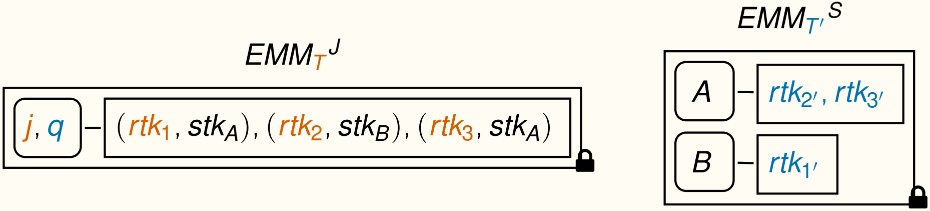


$EMM_T{}^R$

1	$\cdot, E_K(A), \cdot$
2	$\cdot, E_K(B), \cdot$
3	$\cdot, E_K(A), \cdot$

$EMM_{T'}{}^R$

$1'$	$\cdot, E_K(B), \cdot$
$2'$	$\cdot, E_K(A), \cdot$
$3'$	$\cdot, E_K(A), \cdot$



$EMM_T{}^J$

j, q	$(rtk_1, stk_A), (rtk_2, stk_B), (rtk_3, stk_A)$

$EMM_{T'}{}^S$

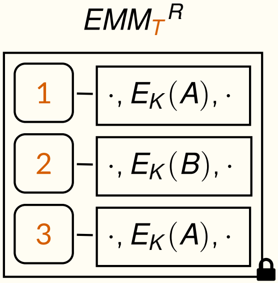
A	$rtk_{2'}, rtk_{3'}$
B	$rtk_{1'}$

Optimal Encrypted Join

$Query(EDB, jtk)$

$\mathcal{O}(T)$

```
For $(rtk, stk)$ in $Query({EMM_T}^J, jtk)$
    $r \leftarrow Query({EMM_T}^R, rtk)$
    For $rtk'$ in $Query({EMM_{T'}}^S, stk)$
        $R' \leftarrow R' \bigcup Query({EMM_{T'}}^R, rtk')$
    Output $(r, R')$
```

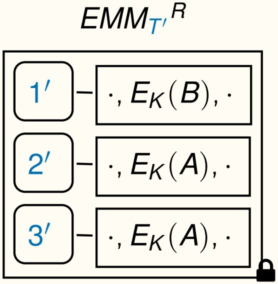


${EMM_T}^R$

1 — $\cdot, E_K(A), \cdot$

2 — $\cdot, E_K(B), \cdot$

3 — $\cdot, E_K(A), \cdot$

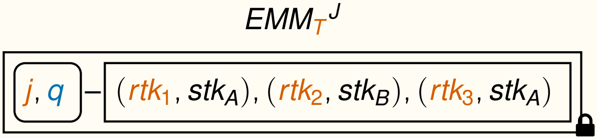


${EMM_{T'}}^R$

$1'$ — $\cdot, E_K(B), \cdot$

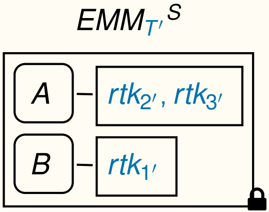
$2'$ — $\cdot, E_K(A), \cdot$

$3'$ — $\cdot, E_K(A), \cdot$



${EMM_T}^J$

j, q — $(rtk_1, stk_A), (rtk_2, stk_B), (rtk_3, stk_A)$



${EMM_{T'}}^S$

A — $rtk_{2'}, rtk_{3'}$

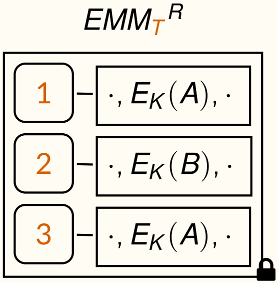
B — $rtk_{1'}$

Optimal Encrypted Join

$Query(EDB, jtk)$

$\mathcal{O}(T)$

```
For $(rtk, stk)$ in $Query(EMM_T{}^J, jtk)$
    $r \leftarrow Query(EMM_T{}^R, rtk)$
    For $rtk'$ in $Query(EMM_{T'}{}^S, stk)$
        $R' \leftarrow R' \cup Query(EMM_{T'}{}^R, rtk')$
    Output $(r, R')$
```

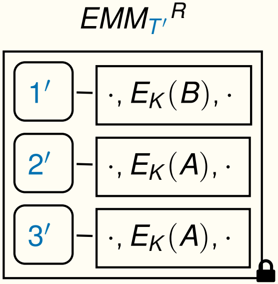


$EMM_T{}^R$

1 – $\cdot, E_K(A), \cdot$

2 – $\cdot, E_K(B), \cdot$

3 – $\cdot, E_K(A), \cdot$

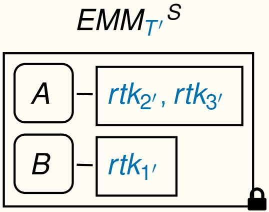


$EMM_{T'}{}^R$

$1'$ – $\cdot, E_K(B), \cdot$

$2'$ – $\cdot, E_K(A), \cdot$

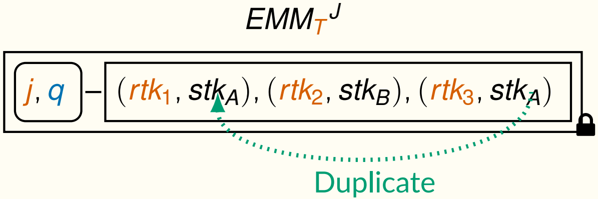
$3'$ – $\cdot, E_K(A), \cdot$



$EMM_{T'}{}^S$

A – $rtk_{2'}, rtk_{3'}$

B – $rtk_{1'}$



$EMM_T{}^J$

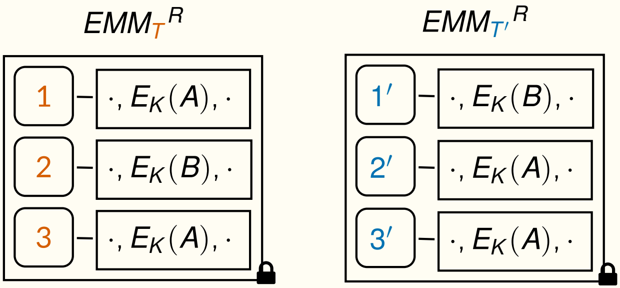
j, q – $(rtk_1, stk_A), (rtk_2, stk_B), (rtk_3, stk_A)$

Duplicate

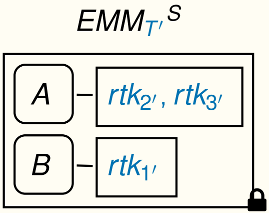
Optimal Encrypted Join

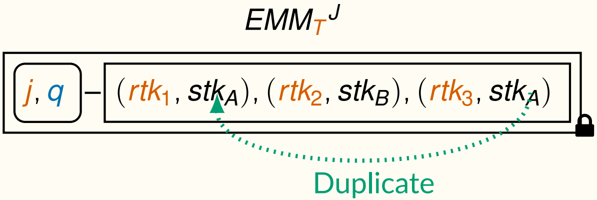
$Query(EDB, jtk)$

```
For $(rtk, stk)$ in $Query(EMM_T{}^J, jtk)$    $\mathcal{O}(T)$
    $r \leftarrow Query(EMM_T{}^R, rtk)$
    // Memoise for same $stk$
    For $rtk'$ in $Query(EMM_{T'}{}^S, stk)$
        $R' \leftarrow R' \cup Query(EMM_{T'}{}^R, rtk')$
    Output $(r, R')$
```



$EMM_T{}^R$
1 — $\cdot, E_K(A), \cdot$
2 — $\cdot, E_K(B), \cdot$
3 — $\cdot, E_K(A), \cdot$
$EMM_{T'}{}^R$
$1'$ — $\cdot, E_K(B), \cdot$
$2'$ — $\cdot, E_K(A), \cdot$
$3'$ — $\cdot, E_K(A), \cdot$

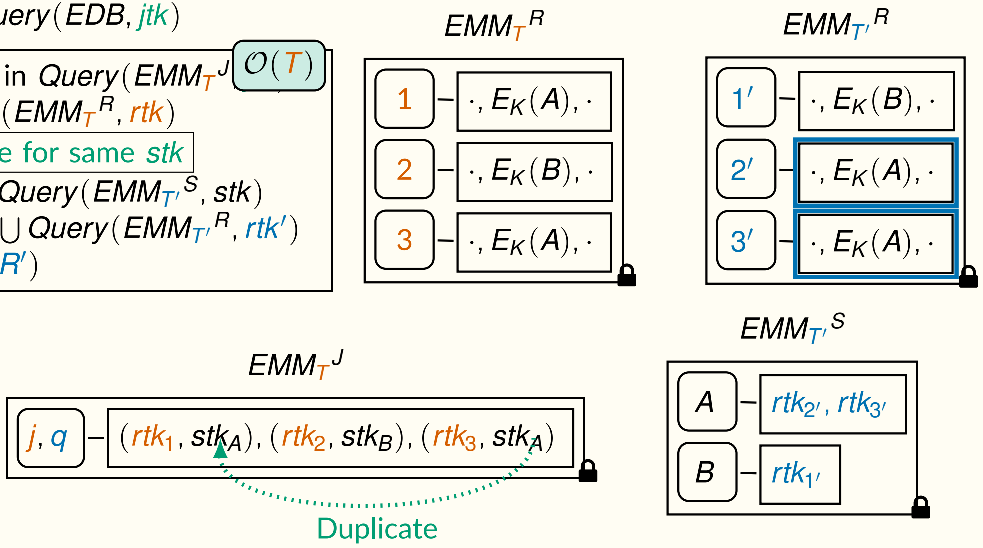




Optimal Encrypted Join

$Query(EDB, jtk)$

```
For $(rtk, stk)$ in $Query(EMM_T{}^J$   $\mathcal{O}(T)$
    $r \leftarrow Query(EMM_T{}^R, rtk)$
    // Memoise for same $stk$
    For $rtk'$ in $Query(EMM_{T'}{}^S, stk)$
        $R' \leftarrow R' \cup Query(EMM_{T'}{}^R, rtk')$
    Output $(r, R')$
```



$EMM_T{}^R$

1	$\cdot, E_K(A), \cdot$
2	$\cdot, E_K(B), \cdot$
3	$\cdot, E_K(A), \cdot$

$EMM_{T'}{}^R$

$1'$	$\cdot, E_K(B), \cdot$
$2'$	$\cdot, E_K(A), \cdot$
$3'$	$\cdot, E_K(A), \cdot$

$EMM_T{}^J$

j, q	$(rtk_1, stk_A), (rtk_2, stk_B), (rtk_3, stk_A)$

Duplicate

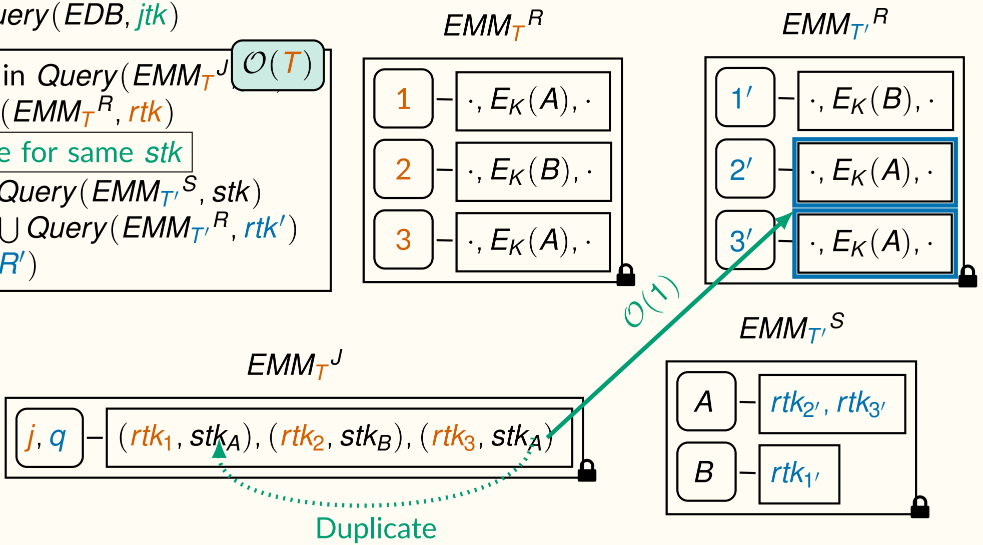
$EMM_{T'}{}^S$

A	$rtk_{2'}, rtk_{3'}$
B	$rtk_{1'}$

Optimal Encrypted Join

$Query(EDB, jtk)$

```
For $(rtk, stk)$ in $Query(EMM_T{}^J$ $\mathcal{O}(T)$
    $r \leftarrow Query(EMM_T{}^R, rtk)$
    // Memoise for same $stk$
    For $rtk'$ in $Query(EMM_{T'}{}^S, stk)$
        $R' \leftarrow R' \cup Query(EMM_{T'}{}^R, rtk')$
    Output $(r, R')$
```



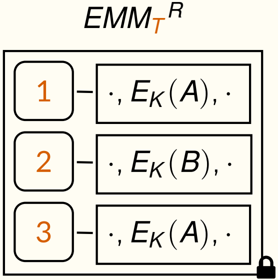
$EMM_T{}^R$
1 — $\cdot, E_K(A), \cdot$
2 — $\cdot, E_K(B), \cdot$
3 — $\cdot, E_K(A), \cdot$
$EMM_{T'}{}^R$
$1'$ — $\cdot, E_K(B), \cdot$
$2'$ — $\cdot, E_K(A), \cdot$
$3'$ — $\cdot, E_K(A), \cdot$
$\mathcal{O}(1)$
$EMM_{T'}{}^S$
A — $rtk_{2'}, rtk_{3'}$
B — $rtk_{1'}$
$EMM_T{}^J$
j, q — $(rtk_1, stk_A), (rtk_2, stk_B), (rtk_3, stk_A)$
Duplicate

Optimal Encrypted Join

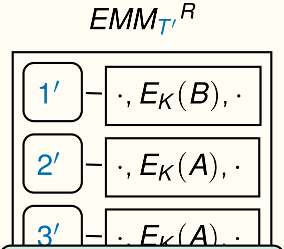
$\mathit{Query}(\mathit{EDB}, \mathit{jtk})$

```
For $(rtk, stk)$ in $\mathit{Query}(\mathit{EMM}_T{}^J$
    $r \leftarrow \mathit{Query}(\mathit{EMM}_T{}^R, rtk)$
    // Memoise for same $stk$
    For $rtk'$ in $\mathit{Query}(\mathit{EMM}_{T'}{}^S, stk)$
        $R' \leftarrow R' \cup \mathit{Query}(\mathit{EMM}_{T'}{}^R, rtk')$
    Output $(r, R')$
```

$\mathcal{O}(T)$

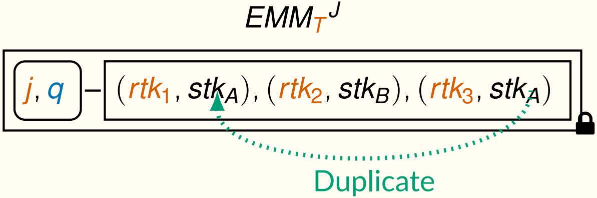


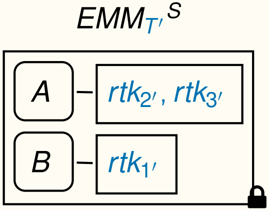
$\mathit{EMM}_T{}^R$
1 — $\cdot, E_K(A), \cdot$
2 — $\cdot, E_K(B), \cdot$
3 — $\cdot, E_K(A), \cdot$



$\mathit{EMM}_{T'}{}^R$
$1'$ — $\cdot, E_K(B), \cdot$
$2'$ — $\cdot, E_K(A), \cdot$
$3'$ — $\cdot, E_K(A), \cdot$

$\mathcal{O}(T')$ total accesses



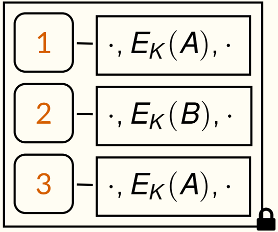


Optimal Encrypted Join

$Query(EDB, jtk)$ $\boxed{\mathcal{O}(T + T')}$

```
For $(rtk, stk)$ in $Query(EMM_T{}^J, jtk)$
    $r \leftarrow Query(EMM_T{}^R, rtk)$
    // Memoise for same $stk$
    For $rtk'$ in $Query(EMM_{T'}{}^S, stk)$
        $R' \leftarrow R' \cup Query(EMM_{T'}{}^R, rtk')$
    Output $(r, R')$
```

$EMM_T{}^R$

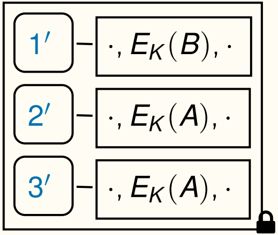


1 — $\cdot, E_K(A), \cdot$

2 — $\cdot, E_K(B), \cdot$

3 — $\cdot, E_K(A), \cdot$

$EMM_{T'}{}^R$

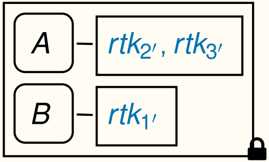


$1'$ — $\cdot, E_K(B), \cdot$

$2'$ — $\cdot, E_K(A), \cdot$

$3'$ — $\cdot, E_K(A), \cdot$

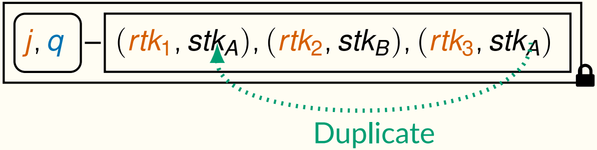
$EMM_{T'}{}^S$



A — $rtk_{2'}, rtk_{3'}$

B — $rtk_{1'}$

$EMM_T{}^J$



j, q — $(rtk_1, stk_A), (rtk_2, stk_B), (rtk_3, stk_A)$

Duplicate

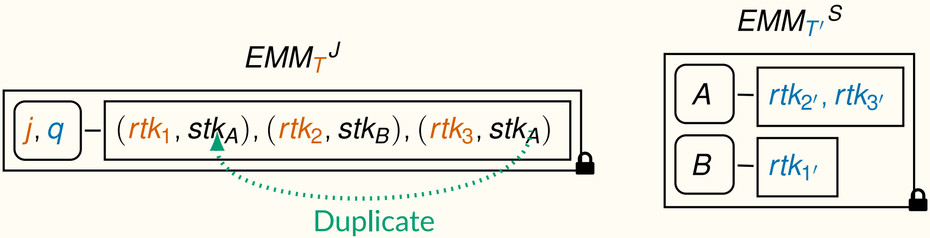
Optimal Encrypted Join

$Query(EDB, jtk$ $\mathcal{O}(T + T')$

```
For $(rtk, stk)$ in $Q$
  $r \leftarrow Query(EM$
  // Memoise for
  For $rtk'$ in $Quer$
    $R' \leftarrow R' \cup Qu$
  Output $(r, R')$
```

$EMM_{T}{}^{R}$ $EMM_{T'}{}^{R}$

- Matches plaintext join asymtotic complexity.
- Improved leakage for conjunctive queries (deferred).



Emulation

Emulation

- Common belief: STE requires re-architecting existing storage system

Emulation

- Common belief: STE requires re-architecting existing storage system
- Emulation: New technique to make STE legacy friendly

Emulation

- Common belief: STE requires re-architecting existing storage system
- Emulation: New technique to make STE legacy friendly
- Express the data structure and language in STE using Relation Model

Emulation

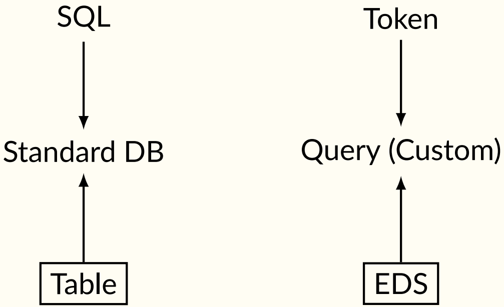
- Common belief: STE requires re-architecting existing storage system
- Emulation: New technique to make STE legacy friendly
- Express the data structure and language in STE using Relation Model
- Does not change security and complexity

Emulation

- Two components: structure reshape and query reform

Emulation

- Two components: structure reshape and query reform



SQL

Standard DB

Table

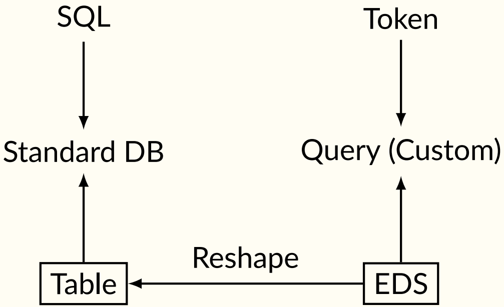
Token

Query (Custom)

EDS

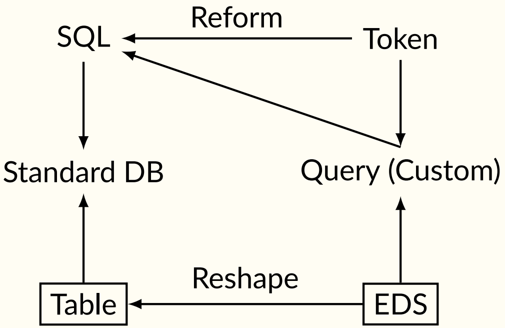
Emulation

- Two components: structure reshape and query reform



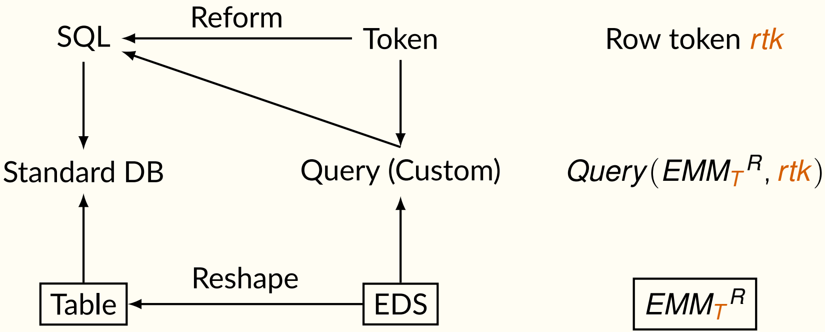
Emulation

- Two components: structure reshape and query reform



Emulation

- Two components: structure reshape and query reform



Reform

SQL

Token

Row token *rtk*

Standard DB

Query (Custom)

$Query(EMM_{T}{}^{R}, rtk)$

Reshape

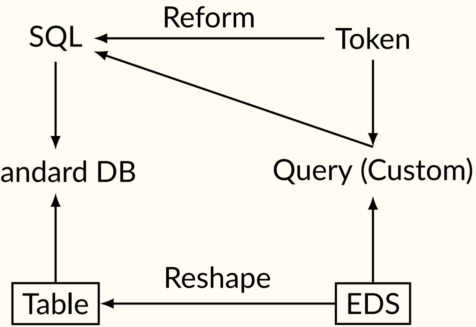
Table

EDS

$EMM_{T}{}^{R}$

Emulation

- Two components: structure reshape and query reform



SQL

Reform

Token

Standard DB

Query (Custom)

Table

Reshape

EDS

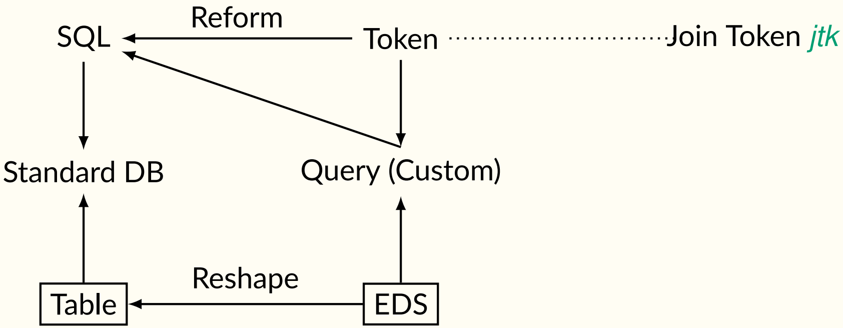
Row token *rtk*

$Query(EMM_T{}^R, rtk)$

$\boxed{EMM_T{}^R}$ — Pibas[CJJ+14]

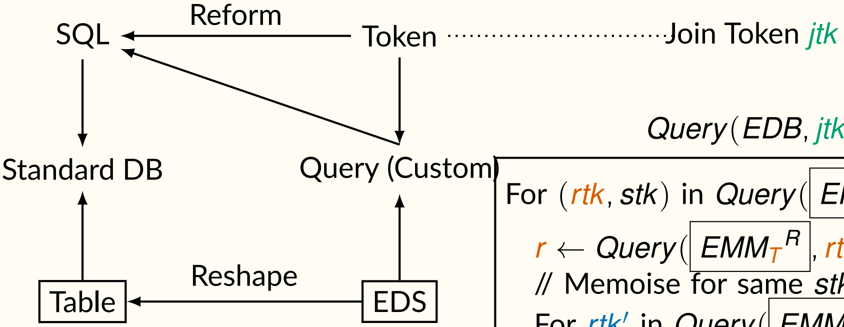
Emulation

- Two components: structure reshape and query reform



Emulation

- Two components: structure reshape and query reform



$Query(EDB, jtk)$

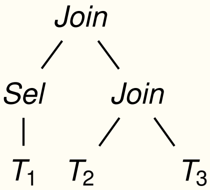
```
For $(rtk, stk)$ in $Query(\boxed{EMM_T{}^J}, jtk)$
    $r \leftarrow Query(\boxed{EMM_T{}^R}, rtk)$
    // Memoise for same $stk$
    For $rtk'$ in $Query(\boxed{EMM_{T'}{}^S}, stk)$
        $R' \leftarrow R' \cup Query(\boxed{EMM_{T'}{}^R}, rtk')$
    Output $(r, R')$
```

Emulation of PKFK Scheme

- PKFK supports conjunctive queries (multiple filters and joins) in SQL

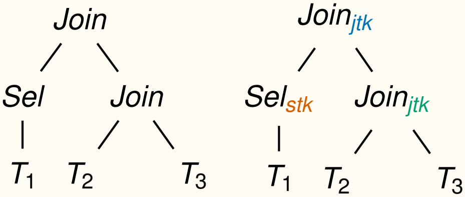
Emulation of PKFK Scheme

- PKFK supports conjunctive queries (multiple filters and joins) in SQL



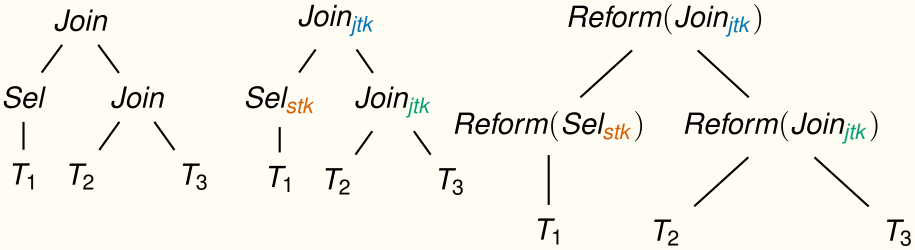
Emulation of PKFK Scheme

- PKFK supports conjunctive queries (multiple filters and joins) in SQL



Emulation of PKFK Scheme

- PKFK supports conjunctive queries (multiple filters and joins) in SQL



Colocation

Colocation

- Data locality is important to achieve high efficiency in database

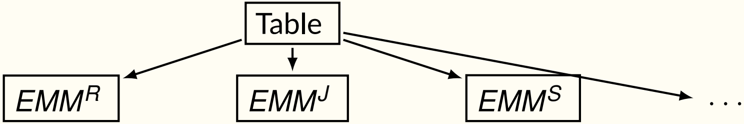
Colocation

- Data locality is important to achieve high efficiency in database

- The representations of EMMs reduces data locality

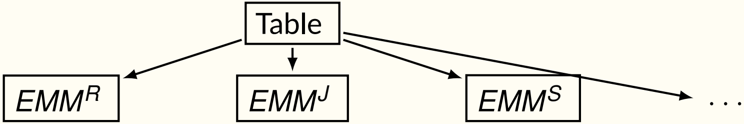
Colocation

- Data locality is important to achieve high efficiency in database
- The representations of EMMs reduces data locality



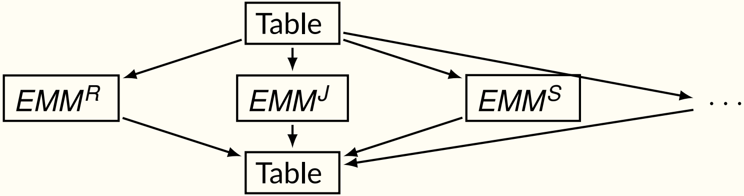
Colocation

- Data locality is important to achieve high efficiency in database
- The representations of EMMs reduces data locality
- Colocation: New technique to increase data locality



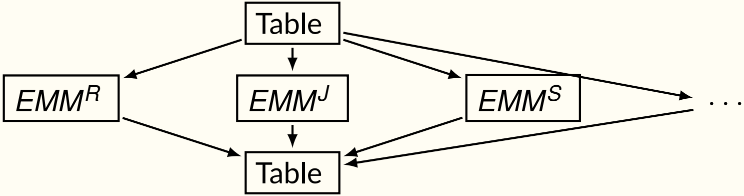
Colocation

- Data locality is important to achieve high efficiency in database
- The representations of EMMs reduces data locality
- Colocation: New technique to increase data locality

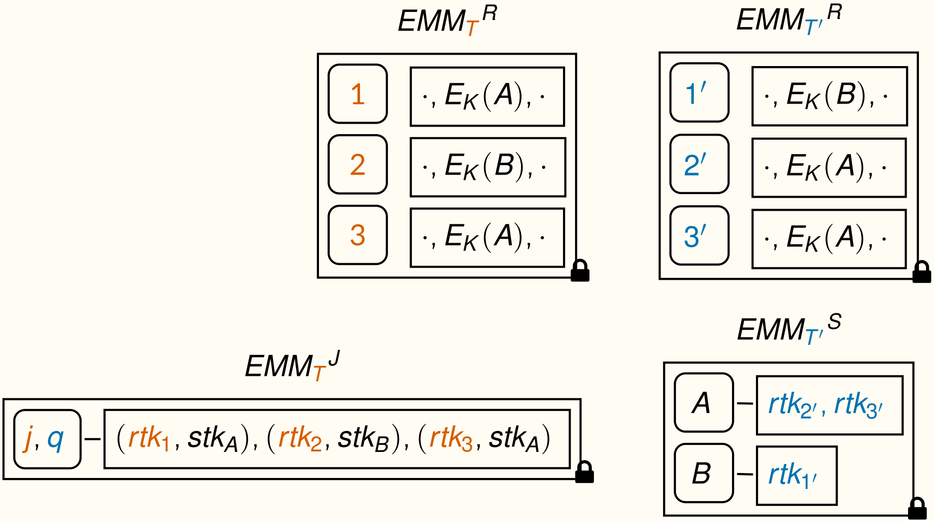


Colocation

- Data locality is important to achieve high efficiency in database
- The representations of EMMs reduces data locality
- Colocation: New technique to increase data locality
- Does not change security



Colocation Example for Join



$EMM_T{}^R$

1	$\cdot, E_K(A), \cdot$
2	$\cdot, E_K(B), \cdot$
3	$\cdot, E_K(A), \cdot$

$EMM_{T'}{}^R$

$1'$	$\cdot, E_K(B), \cdot$
$2'$	$\cdot, E_K(A), \cdot$
$3'$	$\cdot, E_K(A), \cdot$

$EMM_T{}^J$

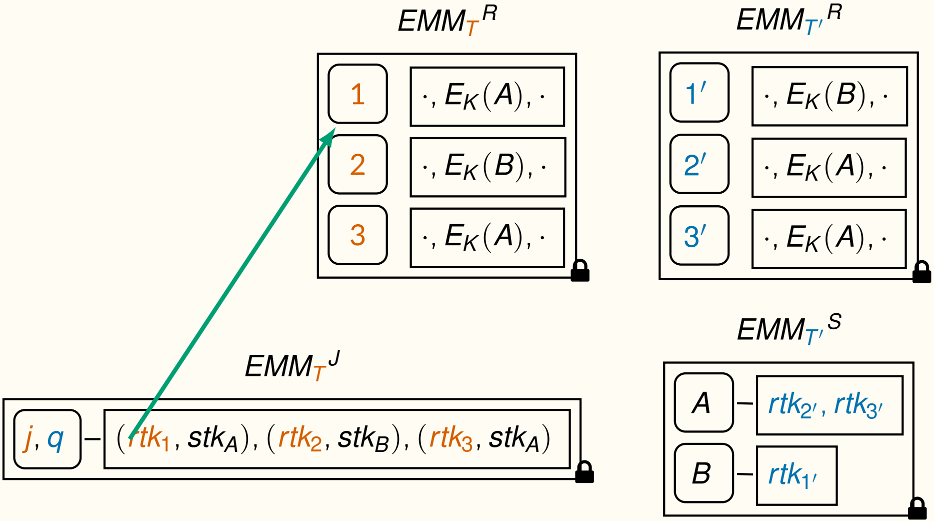
j, q – $(rtk_1, stk_A), (rtk_2, stk_B), (rtk_3, stk_A)$

$EMM_{T'}{}^S$

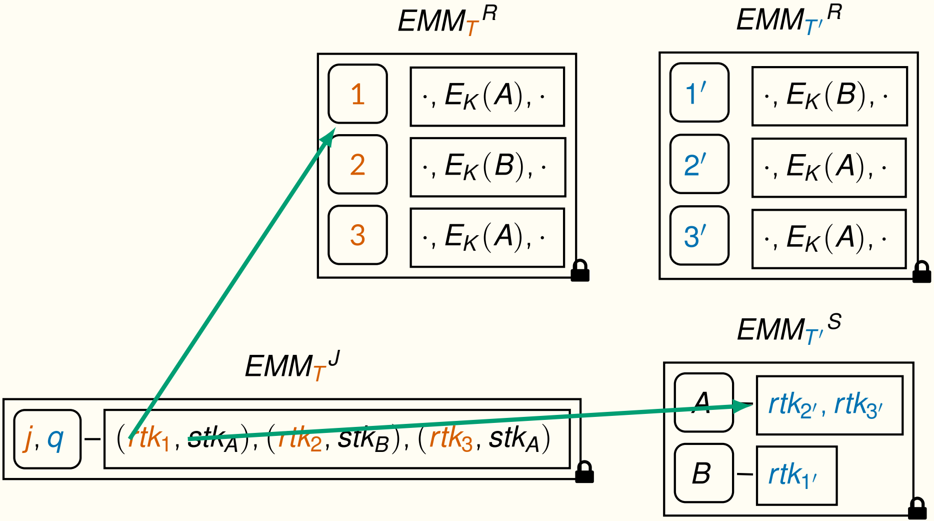
A – $rtk_{2'}, rtk_{3'}$

B – $rtk_{1'}$

Colocation Example for Join



Colocation Example for Join



$EMM_T{}^R$

1 — $\cdot, E_K(A), \cdot$

2 — $\cdot, E_K(B), \cdot$

3 — $\cdot, E_K(A), \cdot$

$EMM_{T'}{}^R$

$1'$ — $\cdot, E_K(B), \cdot$

$2'$ — $\cdot, E_K(A), \cdot$

$3'$ — $\cdot, E_K(A), \cdot$

$EMM_T{}^J$

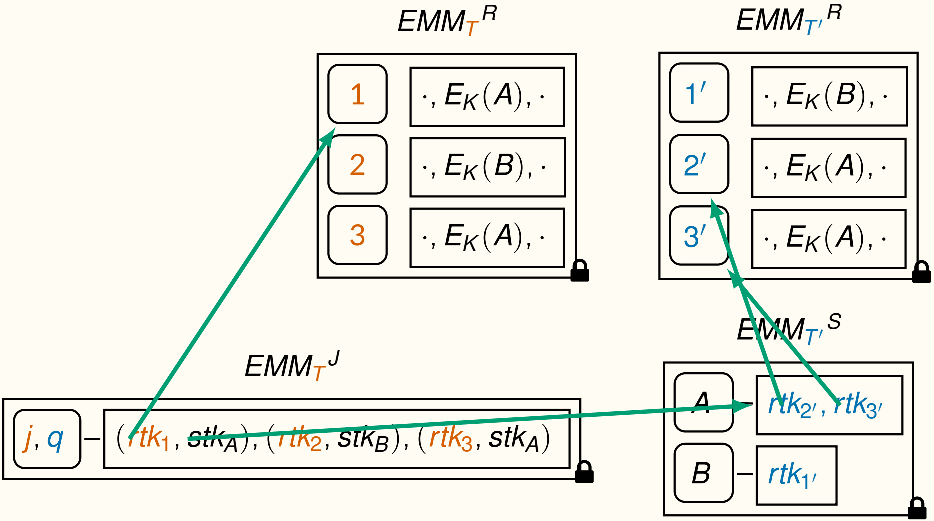
j, q — $(rtk_1, stk_A), (rtk_2, stk_B), (rtk_3, stk_A)$

$EMM_{T'}{}^S$

A — $rtk_{2'}, rtk_{3'}$

B — $rtk_{1'}$

Colocation Example for Join



${EMM_T}^R$

1 — $\cdot, E_K(A), \cdot$

2 — $\cdot, E_K(B), \cdot$

3 — $\cdot, E_K(A), \cdot$

${EMM_{T'}}^R$

$1'$ — $\cdot, E_K(B), \cdot$

$2'$ — $\cdot, E_K(A), \cdot$

$3'$ — $\cdot, E_K(A), \cdot$

${EMM_T}^J$

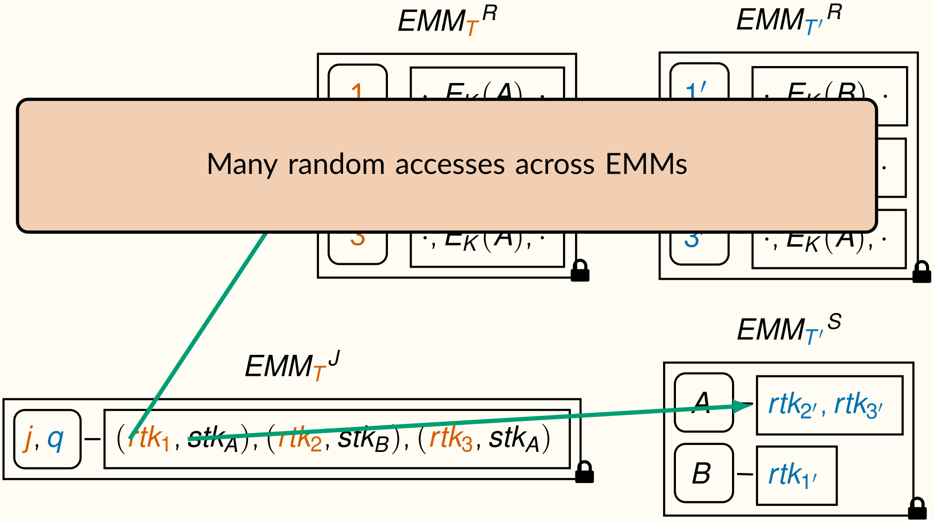
j, q — $(rtk_1, stk_A), (rtk_2, stk_B), (rtk_3, stk_A)$

${EMM_{T'}}^S$

A — $rtk_{2'}, rtk_{3'}$

B — $rtk_{1'}$

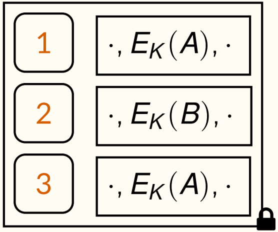
Colocation Example for Join



Many random accesses across EMMs

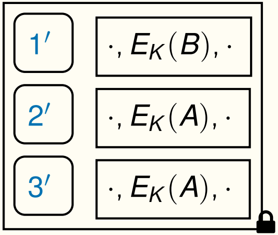
Colocation Example for Join

$EMM_{T}{}^{R}$



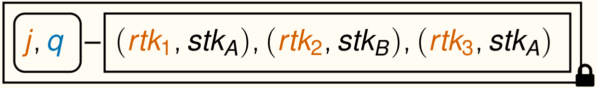
1	$\cdot, E_K(A), \cdot$
2	$\cdot, E_K(B), \cdot$
3	$\cdot, E_K(A), \cdot$

$EMM_{T'}{}^{R}$



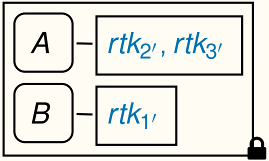
$1'$	$\cdot, E_K(B), \cdot$
$2'$	$\cdot, E_K(A), \cdot$
$3'$	$\cdot, E_K(A), \cdot$

$EMM_{T}{}^{J}$



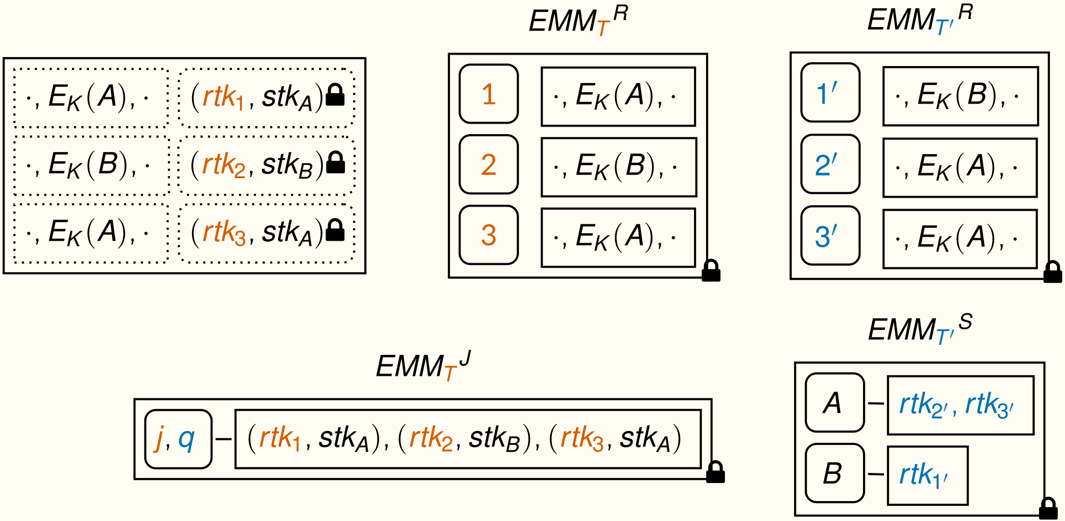
j, q	$(rtk_1, stk_A), (rtk_2, stk_B), (rtk_3, stk_A)$

$EMM_{T'}{}^{S}$



A	$rtk_{2'}, rtk_{3'}$
B	$rtk_{1'}$

Colocation Example for Join



$\cdot, E_K(A), \cdot$ | (rtk_1, stk_A)

$\cdot, E_K(B), \cdot$ | (rtk_2, stk_B)

$\cdot, E_K(A), \cdot$ | (rtk_3, stk_A)

$EMM_T{}^R$

1 | $\cdot, E_K(A), \cdot$

2 | $\cdot, E_K(B), \cdot$

3 | $\cdot, E_K(A), \cdot$

$EMM_{T'}{}^R$

$1'$ | $\cdot, E_K(B), \cdot$

$2'$ | $\cdot, E_K(A), \cdot$

$3'$ | $\cdot, E_K(A), \cdot$

$EMM_T{}^J$

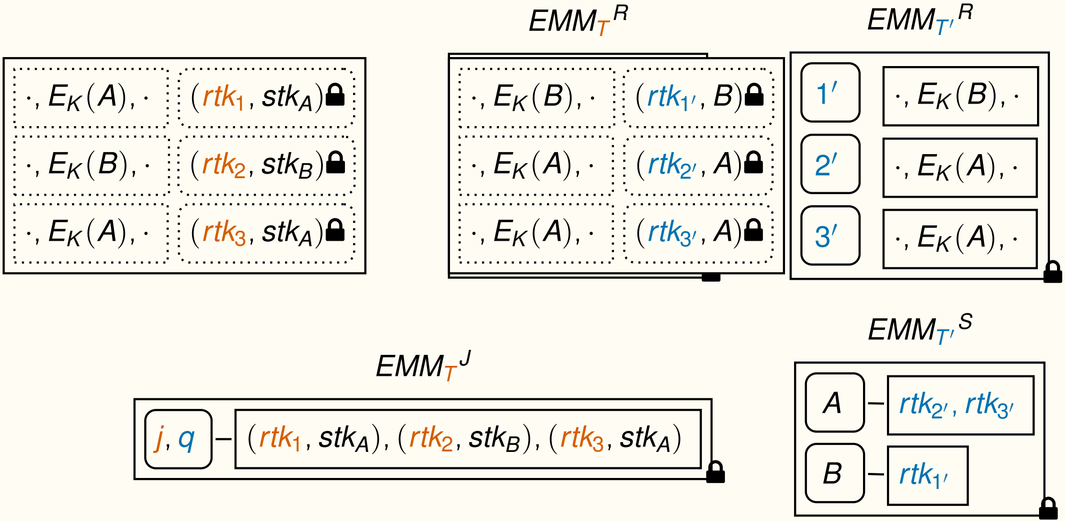
j, q – $(rtk_1, stk_A), (rtk_2, stk_B), (rtk_3, stk_A)$

$EMM_{T'}{}^S$

A – $rtk_{2'}, rtk_{3'}$

B – $rtk_{1'}$

Colocation Example for Join



$\cdot, E_K(A), \cdot$	(rtk_1, stk_A)🔒
$\cdot, E_K(B), \cdot$	(rtk_2, stk_B)🔒
$\cdot, E_K(A), \cdot$	(rtk_3, stk_A)🔒

$EMM_T{}^R$

$\cdot, E_K(B), \cdot$	$(rtk_{1'}, B)$🔒
$\cdot, E_K(A), \cdot$	$(rtk_{2'}, A)$🔒
$\cdot, E_K(A), \cdot$	$(rtk_{3'}, A)$🔒

$EMM_{T'}{}^R$

$1'$	$\cdot, E_K(B), \cdot$
$2'$	$\cdot, E_K(A), \cdot$
$3'$	$\cdot, E_K(A), \cdot$

$EMM_T{}^J$

j, q	$(rtk_1, stk_A), (rtk_2, stk_B), (rtk_3, stk_A)$

$EMM_{T'}{}^S$

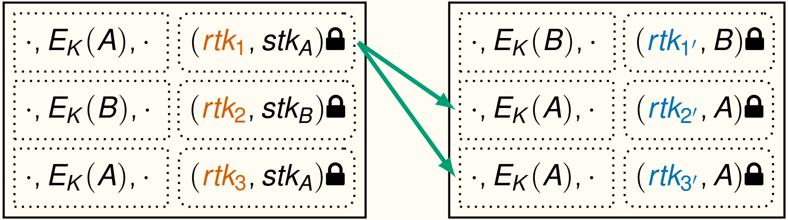
A	$rtk_{2'}, rtk_{3'}$
B	$rtk_{1'}$

Colocation Example for Join

$\cdot, E_K(A), \cdot$	(rtk_1, stk_A)🔒
$\cdot, E_K(B), \cdot$	(rtk_2, stk_B)🔒
$\cdot, E_K(A), \cdot$	(rtk_3, stk_A)🔒

$\cdot, E_K(B), \cdot$	$(rtk_{1'}, B)$🔒
$\cdot, E_K(A), \cdot$	$(rtk_{2'}, A)$🔒
$\cdot, E_K(A), \cdot$	$(rtk_{3'}, A)$🔒

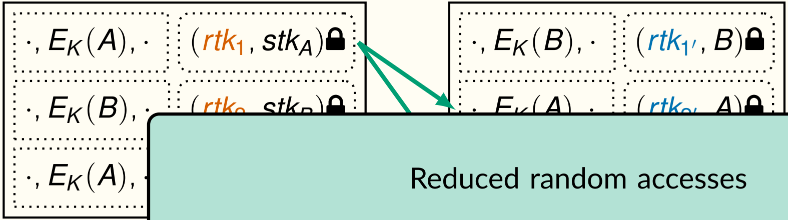
Colocation Example for Join



$\cdot, E_K(A), \cdot$	(rtk_1, stk_A)🔒
$\cdot, E_K(B), \cdot$	(rtk_2, stk_B)🔒
$\cdot, E_K(A), \cdot$	(rtk_3, stk_A)🔒

$\cdot, E_K(B), \cdot$	$(rtk_{1'}, B)$🔒
$\cdot, E_K(A), \cdot$	$(rtk_{2'}, A)$🔒
$\cdot, E_K(A), \cdot$	$(rtk_{3'}, A)$🔒

Colocation Example for Join



Reduced random accesses

The KafeDB System

Legacy-Compatible Architecture, KafeDB

Legacy-Compatible Architecture, KafeDB

- Any SQL database backend

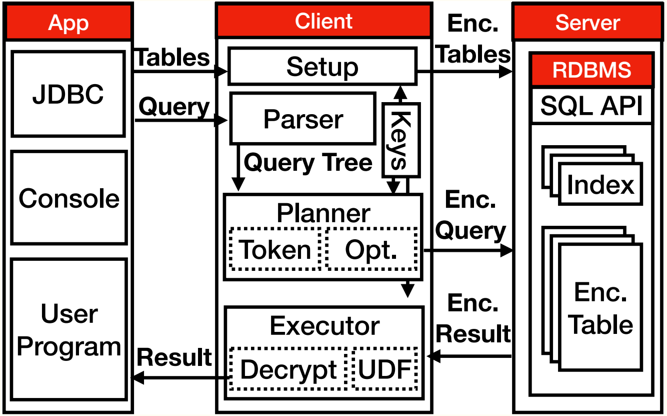
Legacy-Compatible Architecture, KafeDB

- Any SQL database backend
- Applications same API

Legacy-Compatible Architecture, KafeDB

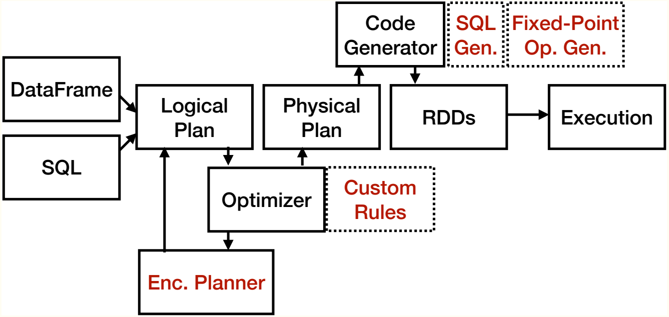
- Any SQL database backend
- Applications same API
- Leverage DB optimizations

Legacy-Compatible Architecture, KafeDB



- Any SQL database backend
- Applications same API
- Leverage DB optimizations

SparkSQL[5]-based Implementation



- PostgreSQL 9.6.2, libcrypto, AES CBC/pkcs7, SHA256.
- BouncyCastle 1.64 on JDK 1.5+
- Apache Spark, Scala, 1000+ lines of code
-

[5] Apache SparkSQL [AXL+15]

Query Optimization Rules

- Important to support query optimization to build upon mature DB research.

Query Optimization Rules

- Important to support query optimization to build upon mature DB research.
- Based on Rules

Query Optimization Rules

- Important to support query optimization to build upon mature DB research.
- Based on Rules

Rules	TPC-H[a]

Query Optimization Rules

- Important to support query optimization to build upon mature DB research.
- Based on Rules
    - Standard Rules: Operator Reorder

Rules	TPC-H[a]

[a] Scale factor 10.

Query Optimization Rules

- Important to support query optimization to build upon mature DB research.
- Based on Rules
  - Standard Rules: Operator Reorder

Rules	TPC-H[a]
Sel/Proj Pushdown	19.8×
Join/Sel Reorder	22.8×

[a] Scale factor 10.

Query Optimization Rules

- Important to support query optimization to build upon mature DB research.
- Based on Rules
  - Standard Rules: Operator Reorder
  - New Rules

Rules	TPC-H[a]
Sel/Proj Pushdown	19.8 ×
Join/Sel Reorder	22.8 ×

[a] Scale factor 10.

Query Optimization Rules

- Important to support query optimization to build upon mature DB research.
- Based on Rules
  - Standard Rules: Operator Reorder
  - New Rules

Rules	TPC-H[a]
Sel/Proj Pushdown	19.8×
Join/Sel Reorder	22.8×
Data Parallel Rewrite[b]	2.7×
Join Direction Reorder	12.6×

[a] Scale factor 10.

Query Optimization Rules

- Important to support query optimization to build upon mature DB research.
- Based on Rules
  - Standard Rules: Operator Reorder
  - New Rules

Rules	TPC-H[a]
Sel/Proj Pushdown	19.8×
Join/Sel Reorder	22.8×
Data Parallel Rewrite[b]	2.7×
Join Direction Reorder	12.6×

[a] Scale factor 10.

[b] 6 cores.

Benchmark

TPC-H Benchmark [TPC-Council'08]

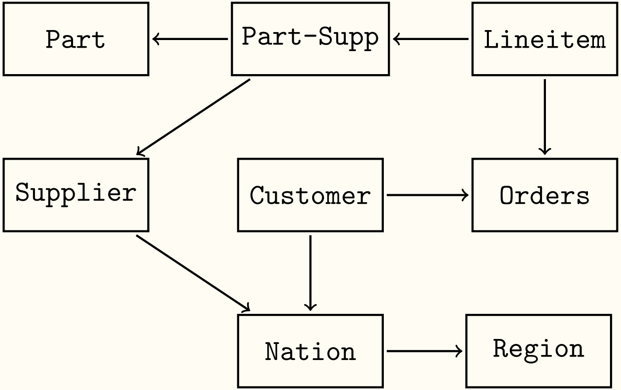
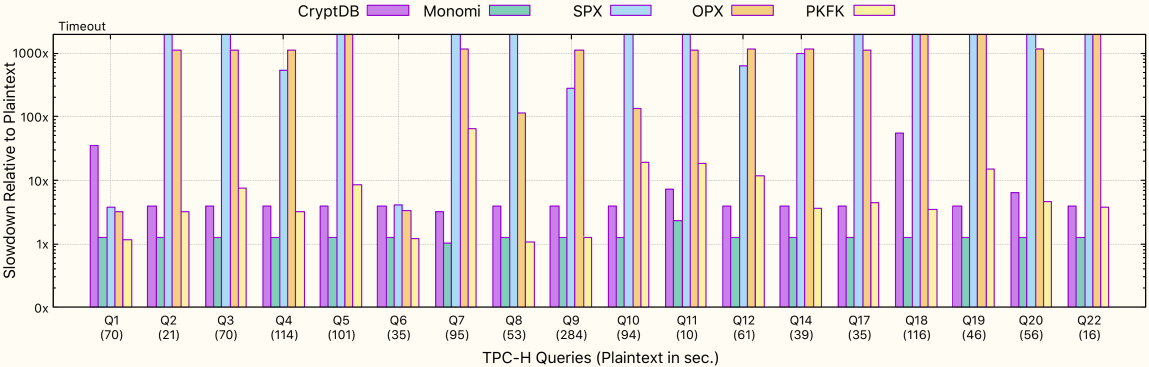


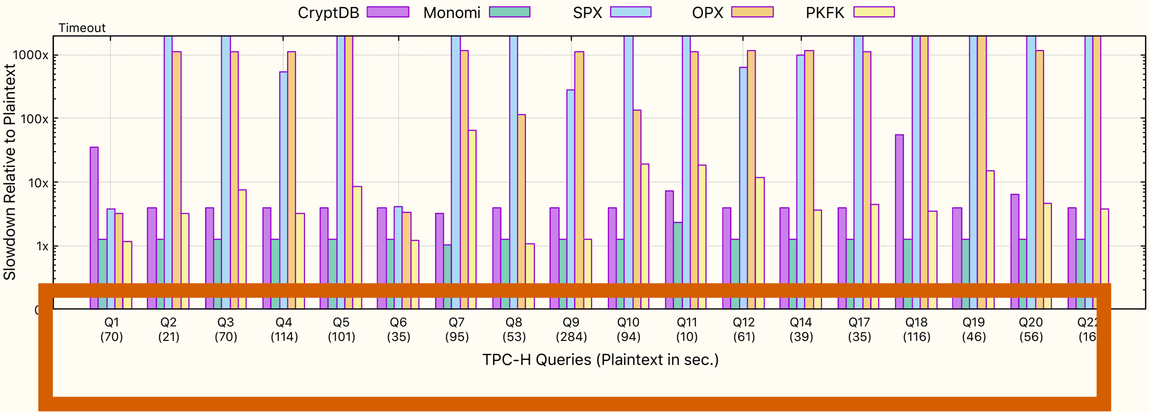
Table	Attrs.	Rows
Part	9	2×10^6
Supplier	7	100×10^3
Part-Supp	7	8×10^6
Customer	8	1.5×10^6
Nation	4	250
Region	3	50
Lineitem	17	60×10^6
Orders	9	15×10^6

- Models data warehouse analyic workload.
- Scale factor 10 (17GB).
- 32GB RAM, 8 CPUs, 5.2TB EBS for SPX and OPX; 1.2TB EBS for PKFK.

Query Overhead



Query Overhead

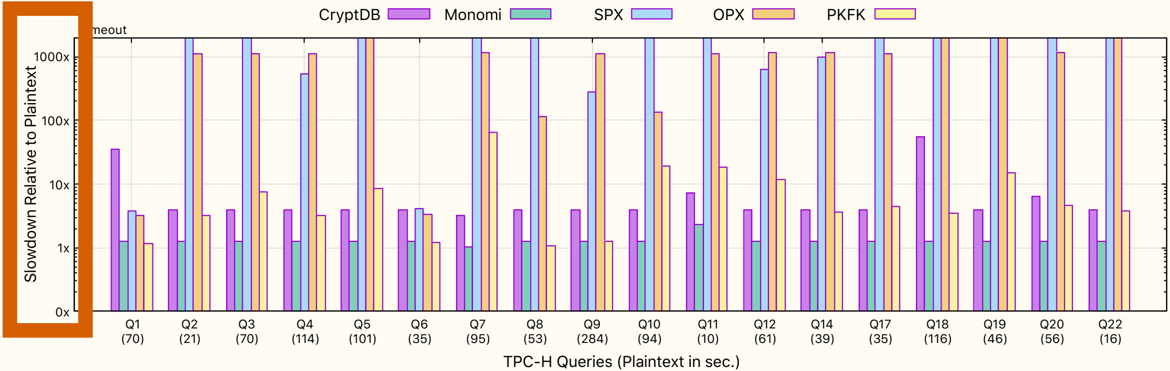


Queries

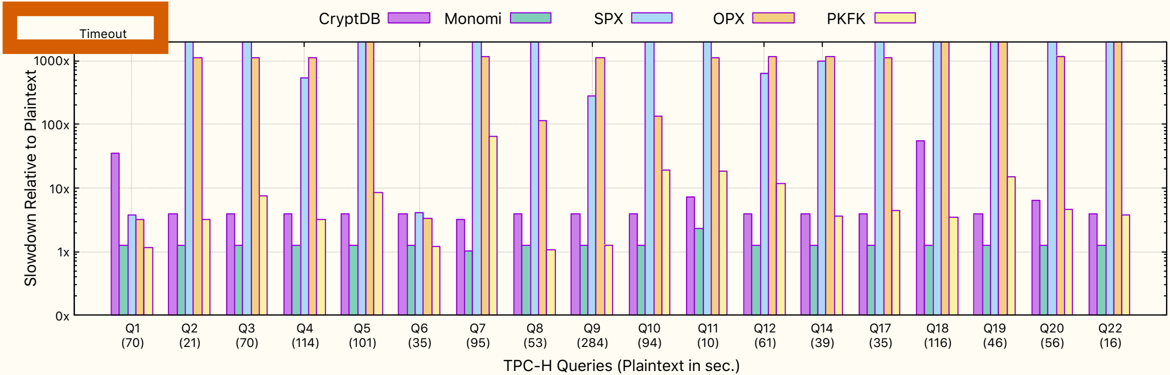
Query Overhead

Overhead

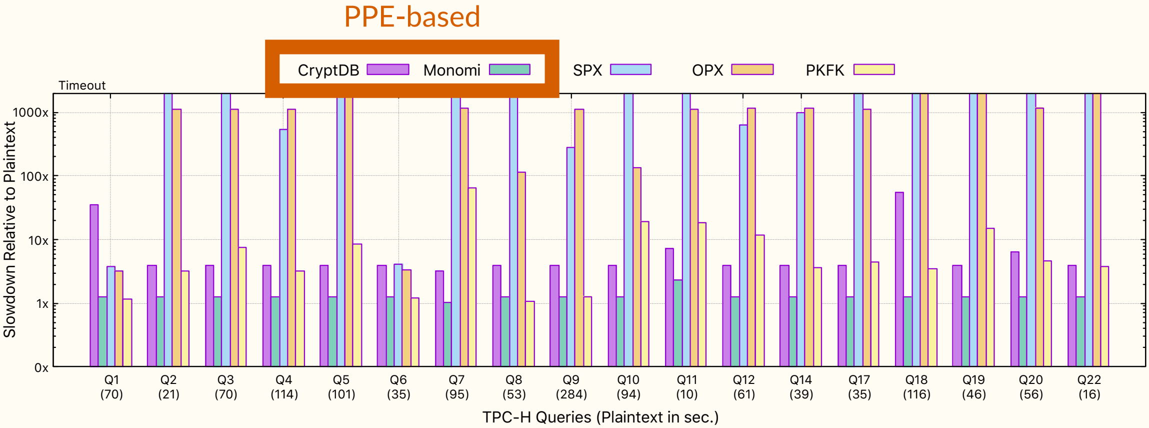
Log Scale



Query Overhead

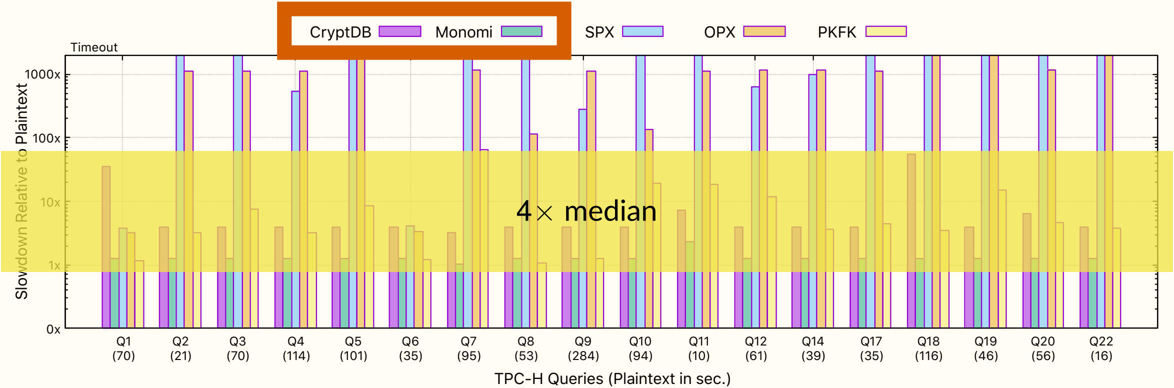


Query Overhead



Query Overhead

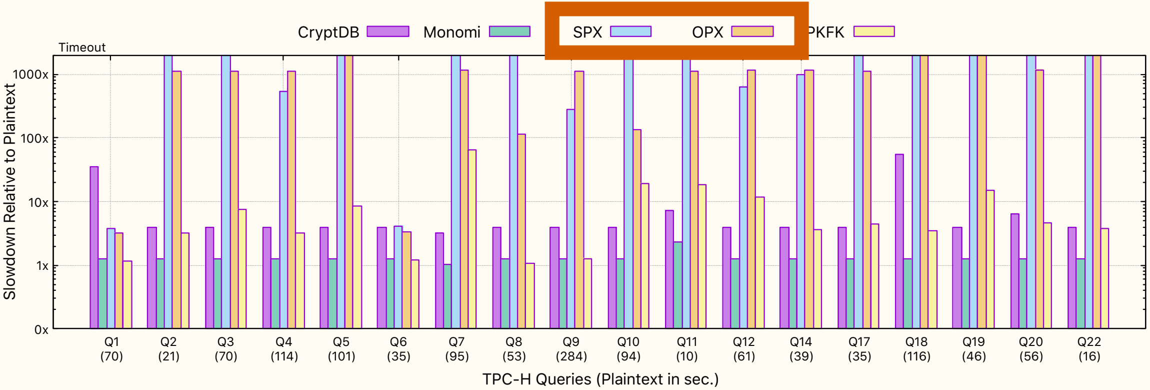
PPE-based



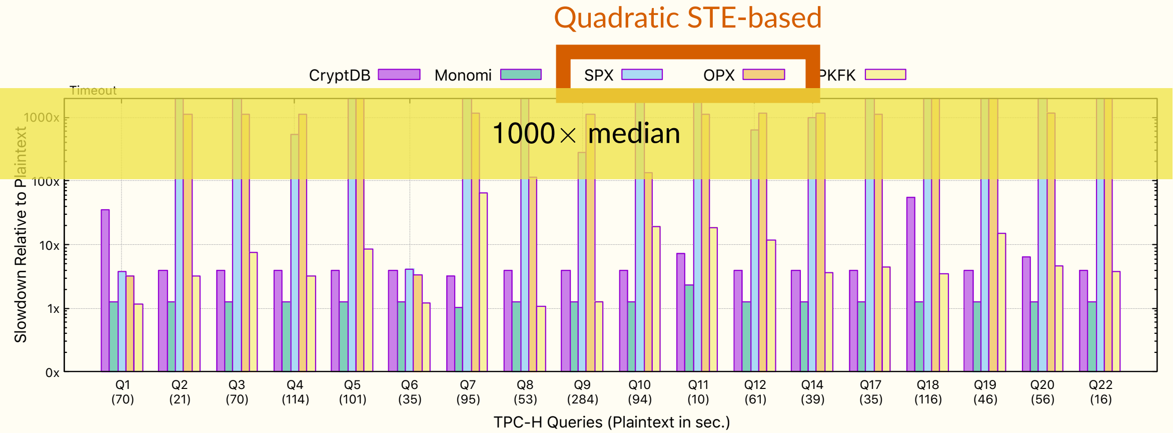
4 × median

Query Overhead

Quadratic STE-based

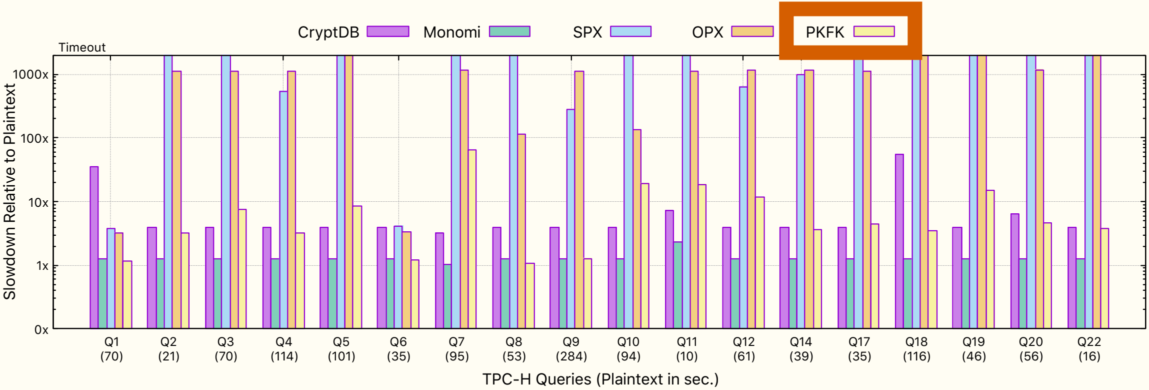


Query Overhead



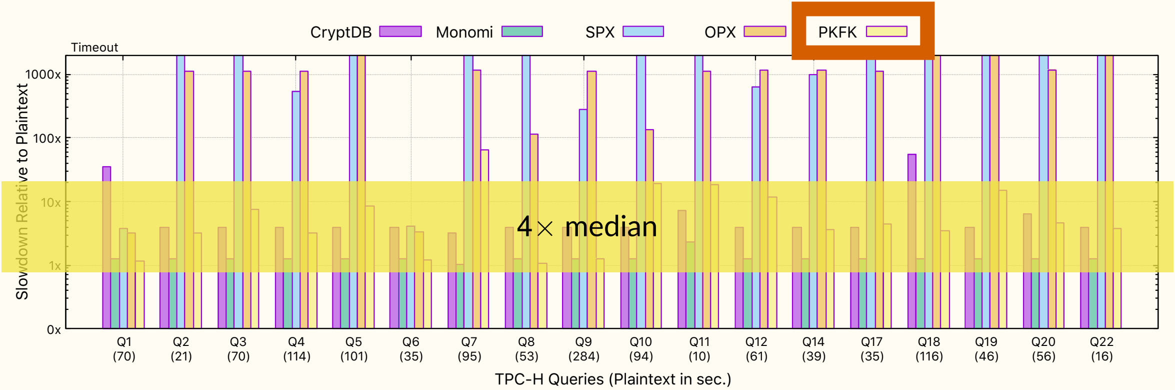
Query Overhead

Optimal STE-based



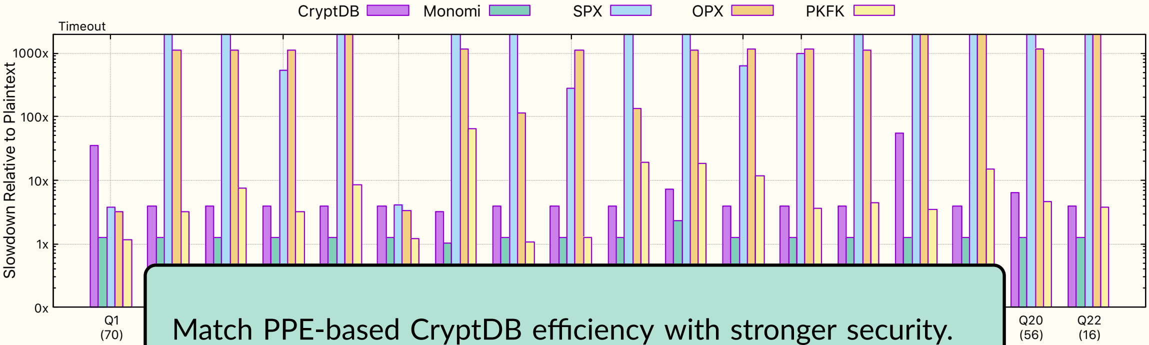
Query Overhead

Optimal STE-based



4 × median

Query Overhead



Match PPE-based CryptDB efficiency with stronger security.

Storage Overhead

System	Size

Storage Overhead

System	Size
Plaintext	17.1GB

Storage Overhead

System	Size
Plaintext	17.1GB
CryptDB	4.2×
Monomi	1.7×

Storage Overhead

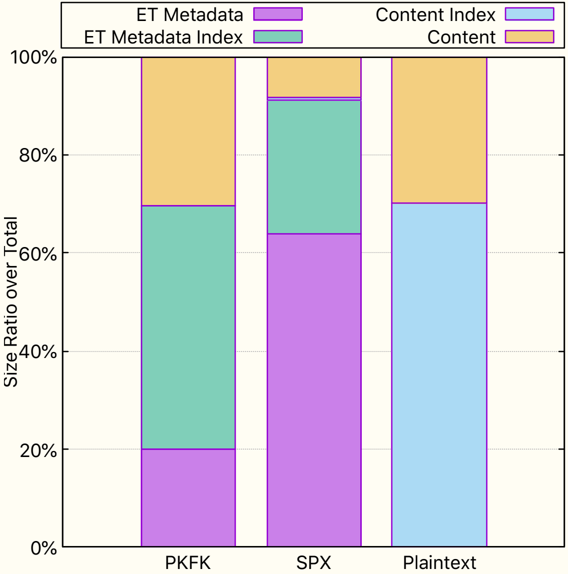
System	Size
Plaintext	17.1GB
CryptDB	4.2×
Monomi	1.7×
SPX	252.2×
OPX	265.4×

Storage Overhead

System	Size
Plaintext	17.1GB
CryptDB	4.2×
Monomi	1.7×
SPX	252.2×
OPX	265.4×
PKFK	3.6×

Storage Overhead

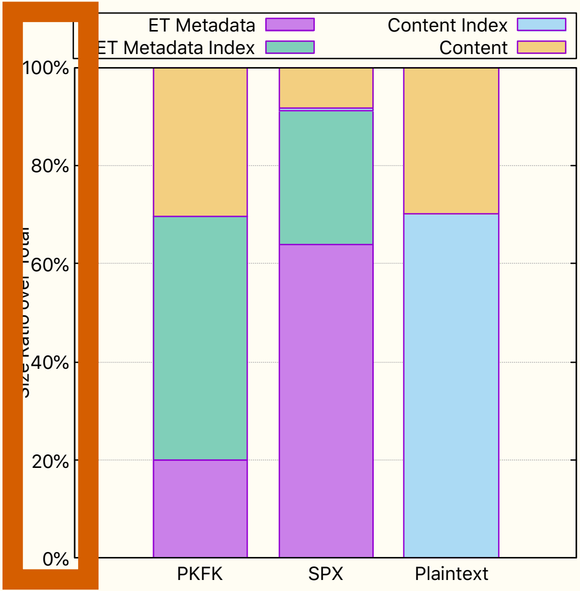
System	Size
Plaintext	17.1GB
CryptDB	4.2×
Monomi	1.7×
SPX	252.2×
OPX	265.4×
PKFK	3.6×



Storage Overhead

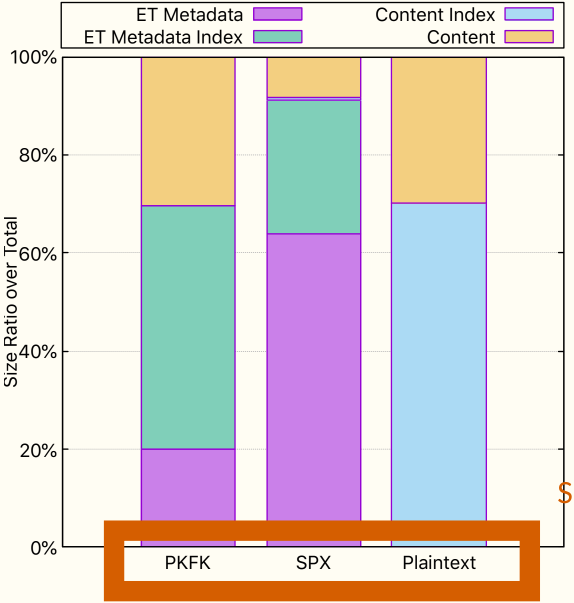
System	Size
Plaintext	17.1GB
CryptDB	4.2×
Monomi	1.7×
SPX	252.2×
OPX	265.4×
PKFK	3.6×

Breakdown Ratio



Storage Overhead

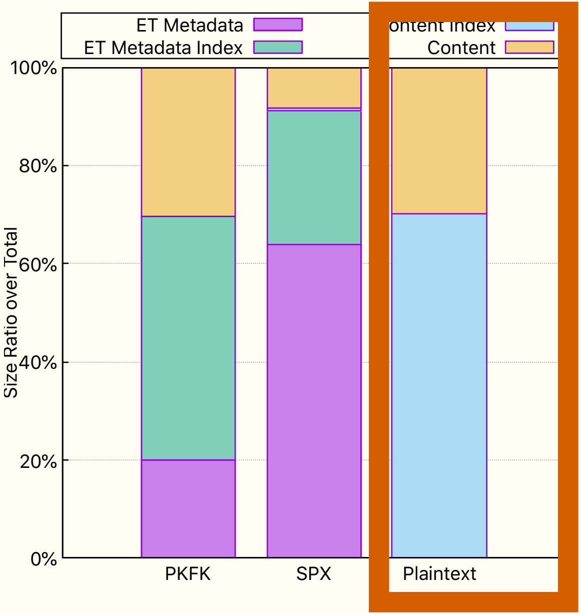
System	Size
Plaintext	17.1GB
CryptDB	4.2×
Monomi	1.7×
SPX	252.2×
OPX	265.4×
PKFK	3.6×



Schemes

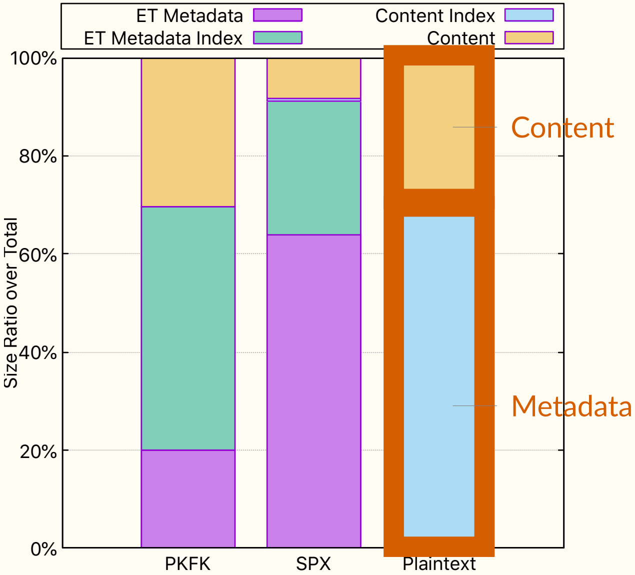
Storage Overhead

System	Size
Plaintext	17.1GB
CryptDB	4.2×
Monomi	1.7×
SPX	252.2×
OPX	265.4×
PKFK	3.6×



Storage Overhead

System	Size
Plaintext	17.1GB
CryptDB	4.2×
Monomi	1.7×
SPX	252.2×
OPX	265.4×
PKFK	3.6×

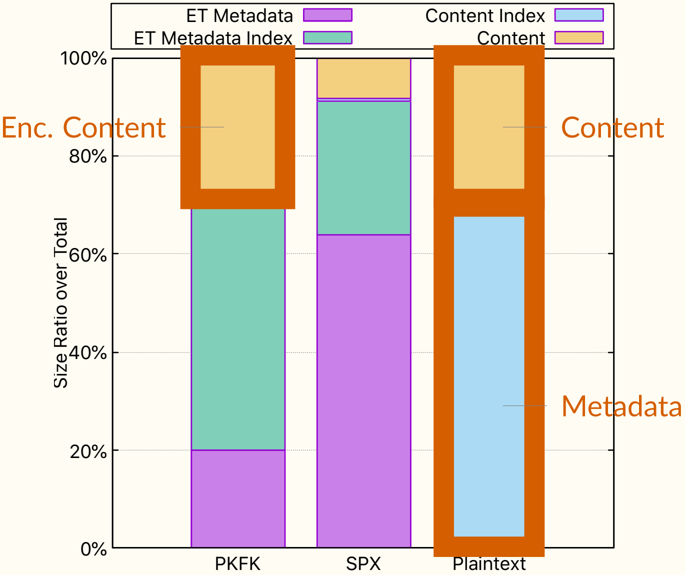


Content

Metadata

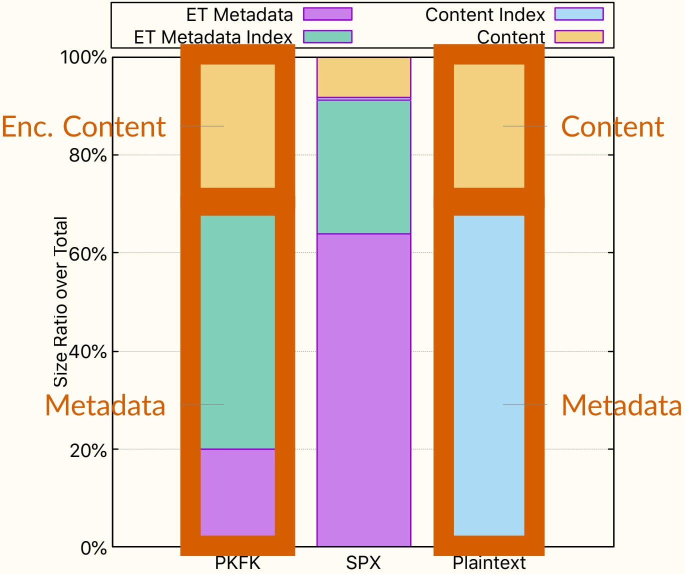
Storage Overhead

System	Size
Plaintext	17.1GB
CryptDB	4.2×
Monomi	1.7×
SPX	252.2×
OPX	265.4×
PKFK	3.6×



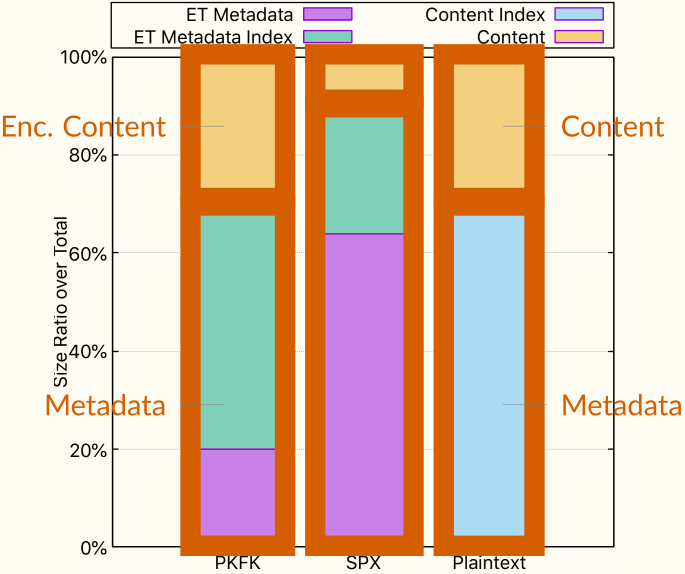
Storage Overhead

System	Size
Plaintext	17.1GB
CryptDB	4.2×
Monomi	1.7×
SPX	252.2×
OPX	265.4×
PKFK	3.6×



Storage Overhead

System	Size
Plaintext	17.1GB
CryptDB	4.2×
Monomi	1.7×
SPX	252.2×
OPX	265.4×
PKFK	3.6×



Enc. Content

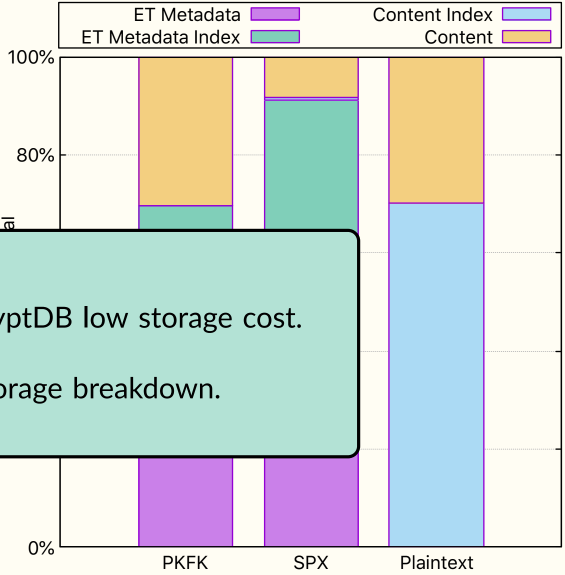
Metadata

Content

Metadata

Storage Overhead

System	Size
Plaintext	17.1GB
CryptDB	4.2×
Monomi	1.7×
SPX	252.2×
OPX	265.4×
PK	



- Match PPE-based CryptDB low storage cost.
- Match Plaintext in storage breakdown.

Summary

- Outsource data securely to the untrusted party such as the cloud.
- STE-based relational scheme: optimal join complexity, expressiveness, improved leakage.
- New techniques: Emulation for legacy. Collocation for locality.
- System based on SparkSQL and interace with any SQL DB.
- Efficiency comparable to PPE-based CryptDB but with stronger security.
- Support for effective query optimization.
- Open source: *https://github.com/zheguang/encrypted-spark*

Appendix

Table vs. Multimap

Data Structure	Table	Multimap
Model	Relational (SQL)	Key-Value (NoSQL)
Language	Relational Algebra	Retrieval by Key
Optimality	$\mathcal{O}(T)$	$\mathcal{O}(Q)$
Basis for EDB	PKFK	SPX,OPX

Encrypted Selection

Name	Pay	Nation
Alice	VISA	US
Bob	AMEX	US
Bob	VISA	CAN

Encrypted Selection

Name	Pay	Nation
Alice$_1$	VISA$_1$	US$_1$
Bob$_1$	AMEX$_1$	US$_2$
Bob$_2$	VISA$_2$	CAN$_1$

Subscripted rep counter

Encrypted Selection

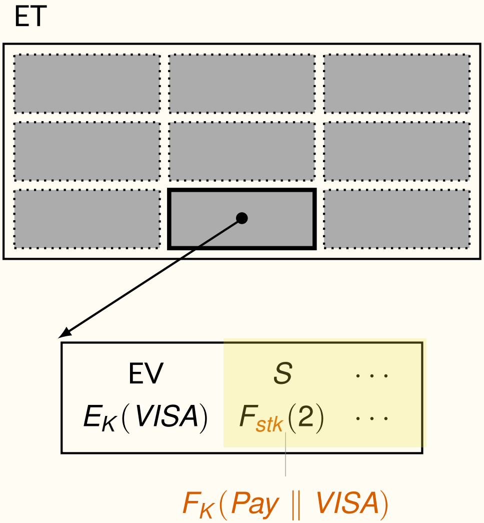
Name	Pay	Nation
Alice_1	VISA_1	US_1
Bob_1	AMEX_1	US_2
Bob_2	VISA_2	CAN_1

ET

▨	▨	▨
▨	▨	▨
▨	▨	▨

Encrypted Selection

Name	Pay	Nation
Alice_1	VISA_1	US_1
Bob_1	AMEX_1	US_2
Bob_2	$\boxed{\text{VISA}_2}$	CAN_1

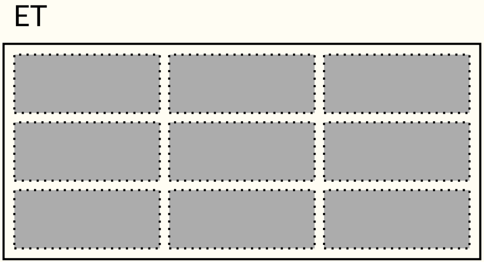


EV	S	$\cdots$
$E_K(VISA)$	$F_{stk}(2)$	$\cdots$

$F_K(Pay \parallel VISA)$

Encrypted Selection

Name	Pay	Nation
Alice$_1$	VISA$_1$	US$_1$
Bob$_1$	AMEX$_1$	US$_2$
Bob$_2$	VISA$_2$	CAN$_1$

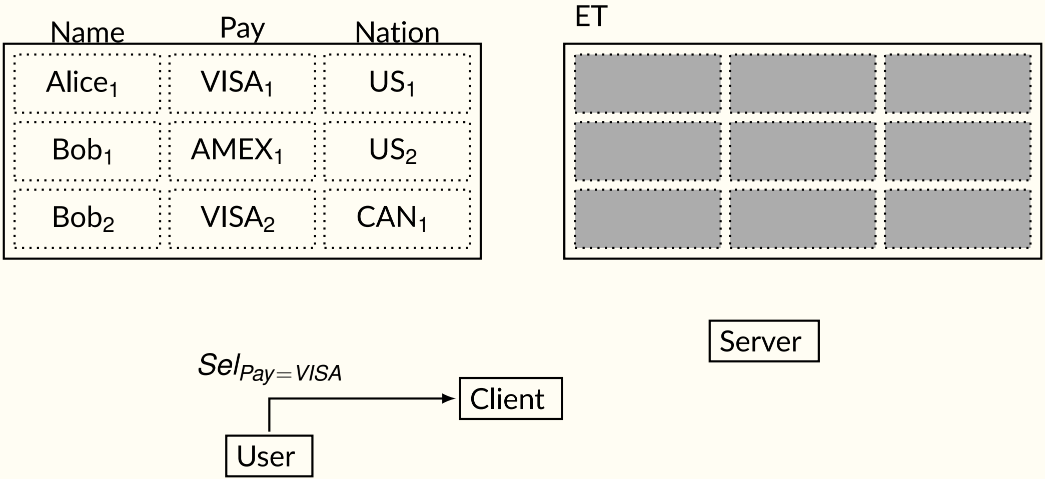


Server

Client

User

Encrypted Selection



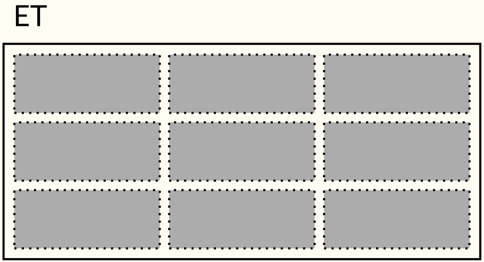
Name	Pay	Nation
Alice_1	VISA_1	US_1
Bob_1	AMEX_1	US_2
Bob_2	VISA_2	CAN_1

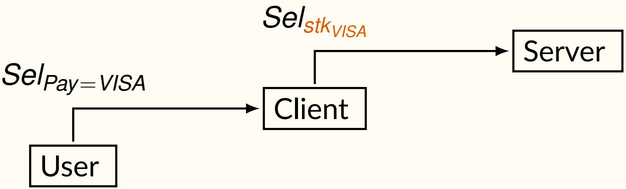
ET

$Sel_{Pay=VISA}$

Encrypted Selection

Name	Pay	Nation
$Alice_1$	$VISA_1$	US_1
Bob_1	$AMEX_1$	US_2
Bob_2	$VISA_2$	CAN_1

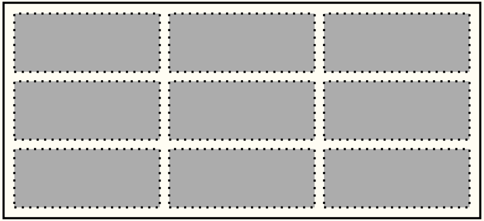


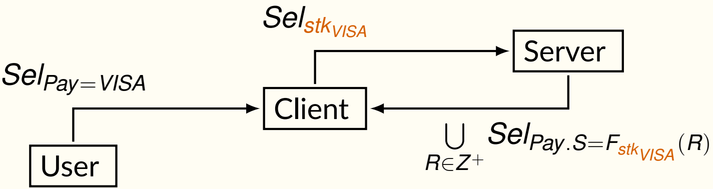


Encrypted Selection

Name	Pay	Nation
Alice_1	VISA_1	US_1
Bob_1	AMEX_1	US_2
Bob_2	VISA_2	CAN_1

ET



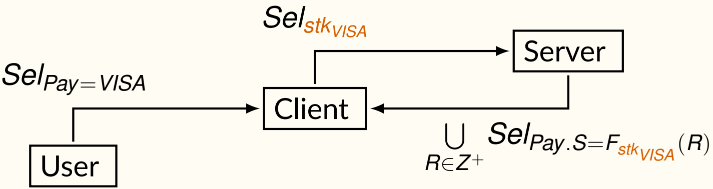


Encrypted Selection

Name	Pay	Nation
$Alice_1$	$VISA_1$	US_1
Bob_1	$AMEX_1$	US_2
Bob_2	$VISA_2$	CAN_1

ET

▨	S	▨
▨	S	▨
▨	S	▨



$Sel_{stk_{VISA}}$

Server

$Sel_{Pay=VISA}$

Client

$\bigcup_{R \in Z^+} Sel_{Pay.S=F_{stk_{VISA}}(R)}$

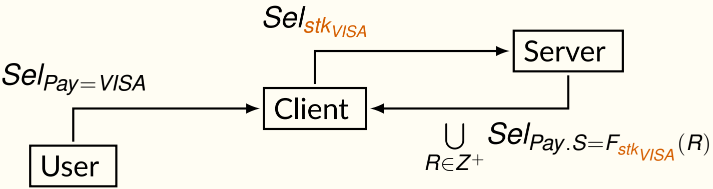
User

Encrypted Selection

Name	Pay	Nation
$Alice_1$	$VISA_1$	US_1
Bob_1	$AMEX_1$	US_2
Bob_2	$VISA_2$	CAN_1

ET

▨	$F_{stk_{VISA}}(1)$	▨
▨	$F_{stk_{AMEX}}(1)$	▨
▨	$F_{stk_{VISA}}(2)$	▨



$Sel_{stk_{VISA}}$

Server

$Sel_{Pay=VISA}$

Client

User

$\bigcup_{R\in Z^+} Sel_{Pay.S=F_{stk_{VISA}}(R)}$

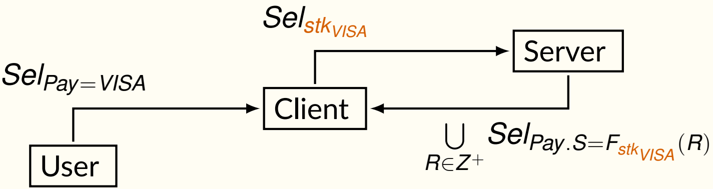
Encrypted Selection

Name	Pay	Nation
Alice_1	VISA_1	US_1
Bob_1	AMEX_1	US_2
Bob_2	VISA_2	CAN_1

ET

Run: $\sigma_{Pay.S=F_{stk_{VISA}}}(1)$

▨	$F_{stk_{VISA}}(1)$	▨
▨	$F_{stk_{AMEX}}(1)$	▨
▨	$F_{stk_{VISA}}(2)$	▨



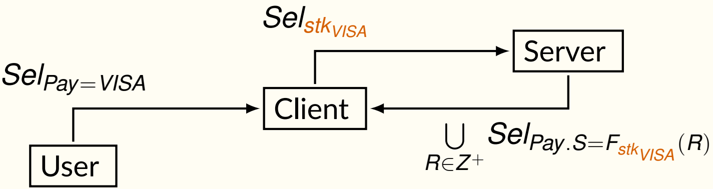
Encrypted Selection

Name	Pay	Nation
Alice_1	VISA_1	US_1
Bob_1	AMEX_1	US_2
Bob_2	VISA_2	CAN_1

ET

Run: $\sigma_{Pay.S=F_{stk_{VISA}}(1)}$

▨	🟩	▨
▨	$F_{stk_{AMEX}}(1)$	▨
▨	$F_{stk_{VISA}}(2)$	▨



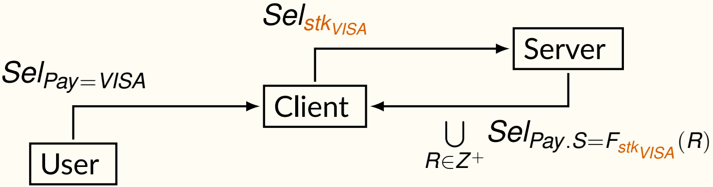
Encrypted Selection

Name	Pay	Nation
Alice_1	$\boxed{\text{VISA}_1}$	US_1
Bob_1	AMEX_1	US_2
Bob_2	VISA_2	CAN_1

ET

Run: $\sigma_{Pay.S=F_{stk_{VISA}}}(2)$

▨	🟩	▨
▨	$F_{stk_{AMEX}}(1)$	▨
▨	$F_{stk_{VISA}}(2)$	▨

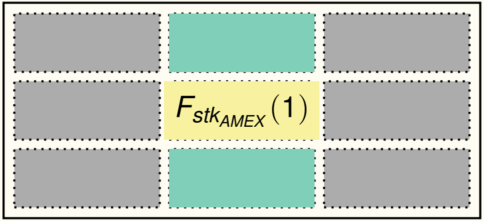


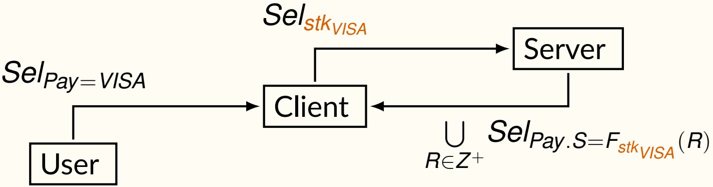
Encrypted Selection

Name	Pay	Nation
Alice$_1$	VISA$_1$	US$_1$
Bob$_1$	AMEX$_1$	US$_2$
Bob$_2$	VISA$_2$	CAN$_1$

ET

Run: $\sigma_{Pay.S=F_{stk_{VISA}}}(2)$





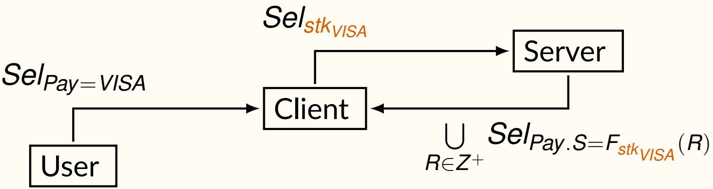
Encrypted Selection

Name	Pay	Nation
Alice_1	VISA_1	US_1
Bob_1	AMEX_1	US_2
Bob_2	VISA_2	CAN_1

ET

Run: $\sigma_{Pay.S=F_{stk_{VISA}}}(3)$

▨	🟩	▨
▨	$F_{stk_{AMEX}}(1)$	▨
▨	🟩	▨

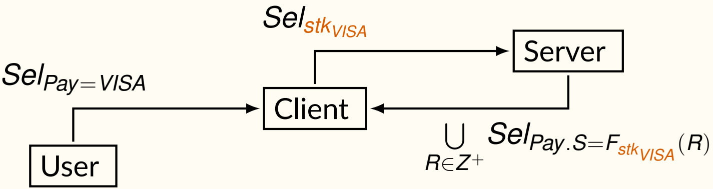


Encrypted Selection

Name	Pay	Nation
Alice$_1$	VISA$_1$	US$_1$
Bob$_1$	AMEX$_1$	US$_2$
Bob$_2$	VISA$_2$	CAN$_1$

ET

	Done	
▨	🟩	▨
▨	$F_{stk_{AMEX}}(1)$	▨
▨	🟩	▨

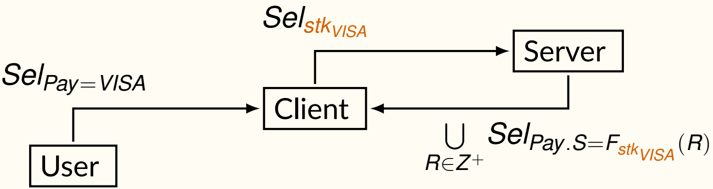


Encrypted Selection

Name	Pay	Nation
$Alice_1$	$VISA_1$	US_1
Bob_1	$AMEX_1$	US_2
Bob_2	$VISA_2$	CAN_1

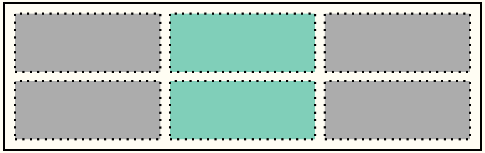
ET

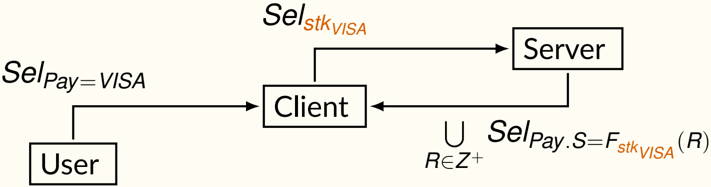
▨	🟩	▨
□	□	□
▨	🟩	▨



Encrypted Selection

Result ET





$Sel_{stk_{VISA}}$

Server

$Sel_{Pay=VISA}$

Client

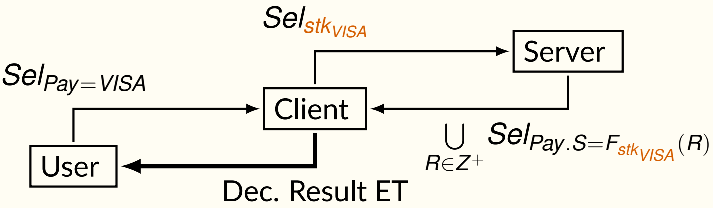
User

$\bigcup_{R \in Z^+} Sel_{Pay.S=F_{stk_{VISA}}(R)}$

Encrypted Selection

Result ET

$D_K(EV)$	$D_K(EV)$	$D_K(EV)$
$D_K(EV)$	$D_K(EV)$	$D_K(EV)$



$Sel_{stk_{VISA}}$

Server

$Sel_{Pay=VISA}$

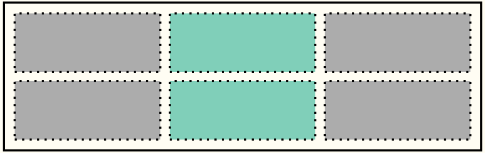
Client

$\bigcup_{R \in Z^+} Sel_{Pay.S=F_{stk_{VISA}}(R)}$

User

Dec. Result ET

Encrypted Selection

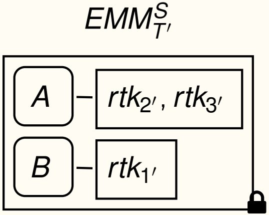


Result ET

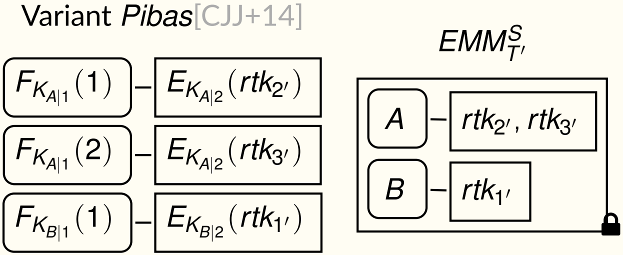
Encrypted Selection

- Leakage: equality pattern in result, etc.
- Sublinear cost.
- Locality.

Emulation of Pibas EMM [CJJ+14]



Emulation of Pibas EMM [CJJ+14]



Variant *Pibas*[CJJ+14]

$F_{K_{A|1}}(1)$ – $E_{K_{A|2}}(rtk_{2'})$

$F_{K_{A|1}}(2)$ – $E_{K_{A|2}}(rtk_{3'})$

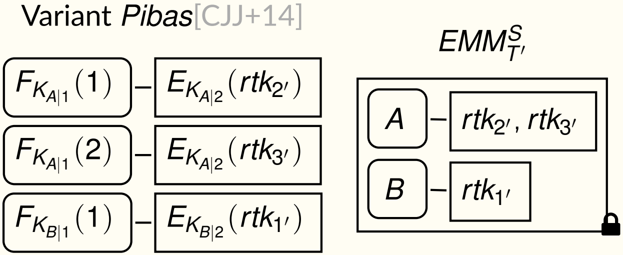
$F_{K_{B|1}}(1)$ – $E_{K_{B|2}}(rtk_{1'})$

$EMM^{S}_{T'}$

A – $rtk_{2'}, rtk_{3'}$

B – $rtk_{1'}$

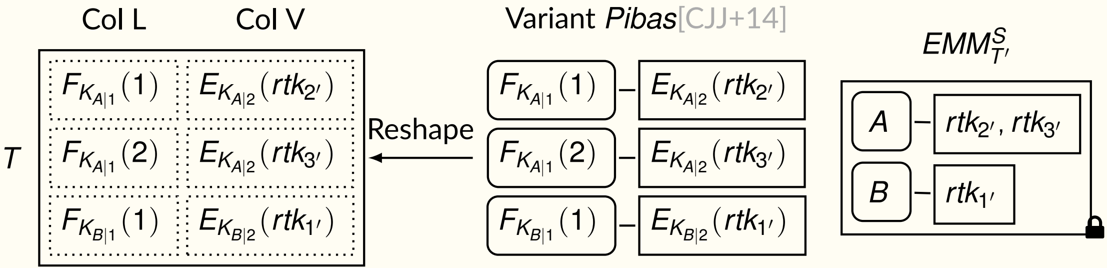
Emulation of Pibas EMM [CJJ+14]


Variant *Pibas*[CJJ+14]
$EMM^S_{T'}$
$F_{K_{A|1}}(1)$ – $E_{K_{A|2}}(rtk_{2'})$
$F_{K_{A|1}}(2)$ – $E_{K_{A|2}}(rtk_{3'})$
$F_{K_{B|1}}(1)$ – $E_{K_{B|2}}(rtk_{1'})$
A – $rtk_{2'}, rtk_{3'}$
B – $rtk_{1'}$

$Query(stk_A)$

```
$K_{A|1} = F_{stk_A}(1)$
$K_{A|2} = F_{stk_A}(2)$
For $R = 1, 2, \cdots$
    $v = Get(F_{K_{A|1}}(R))$
    Output $D_{K_{A|2}}(v)$
```

Emulation of Pibas EMM [CJJ+14]



T

Col L	Col V
$F_{K_{A\|1}}(1)$	$E_{K_{A\|2}}(rtk_{2'})$
$F_{K_{A\|1}}(2)$	$E_{K_{A\|2}}(rtk_{3'})$
$F_{K_{B\|1}}(1)$	$E_{K_{B\|2}}(rtk_{1'})$

Reshape

Variant *Pibas*[CJJ+14]

$F_{K_{A\|1}}(1)$	$E_{K_{A\|2}}(rtk_{2'})$
$F_{K_{A\|1}}(2)$	$E_{K_{A\|2}}(rtk_{3'})$
$F_{K_{B\|1}}(1)$	$E_{K_{B\|2}}(rtk_{1'})$

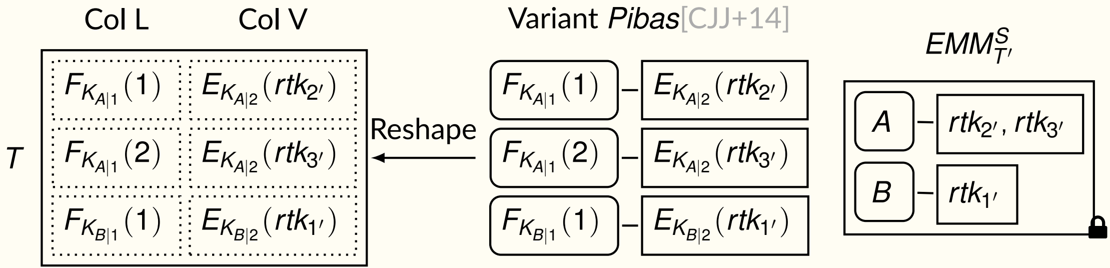
$EMM^{S}_{T'}$

A	$rtk_{2'}, rtk_{3'}$
B	$rtk_{1'}$

$Query(stk_A)$

```
$K_{A|1} = F_{stk_A}(1)$
$K_{A|2} = F_{stk_A}(2)$
For $R = 1, 2, \cdots$
    $v = Get(F_{K_{A|1}}(R))$
    Output $D_{K_{A|2}}(v)$
```

Emulation of Pibas EMM [CJJ+14]



	Col L	Col V
T	$F_{K_{A\|1}}(1)$	$E_{K_{A\|2}}(rtk_{2'})$
	$F_{K_{A\|1}}(2)$	$E_{K_{A\|2}}(rtk_{3'})$
	$F_{K_{B\|1}}(1)$	$E_{K_{B\|2}}(rtk_{1'})$

Reshape

Variant *Pibas*[CJJ+14]

$F_{K_{A|1}}(1)$ — $E_{K_{A|2}}(rtk_{2'})$

$F_{K_{A|1}}(2)$ — $E_{K_{A|2}}(rtk_{3'})$

$F_{K_{B|1}}(1)$ — $E_{K_{B|2}}(rtk_{1'})$

$EMM^S_{T'}$

A — $rtk_{2'}, rtk_{3'}$

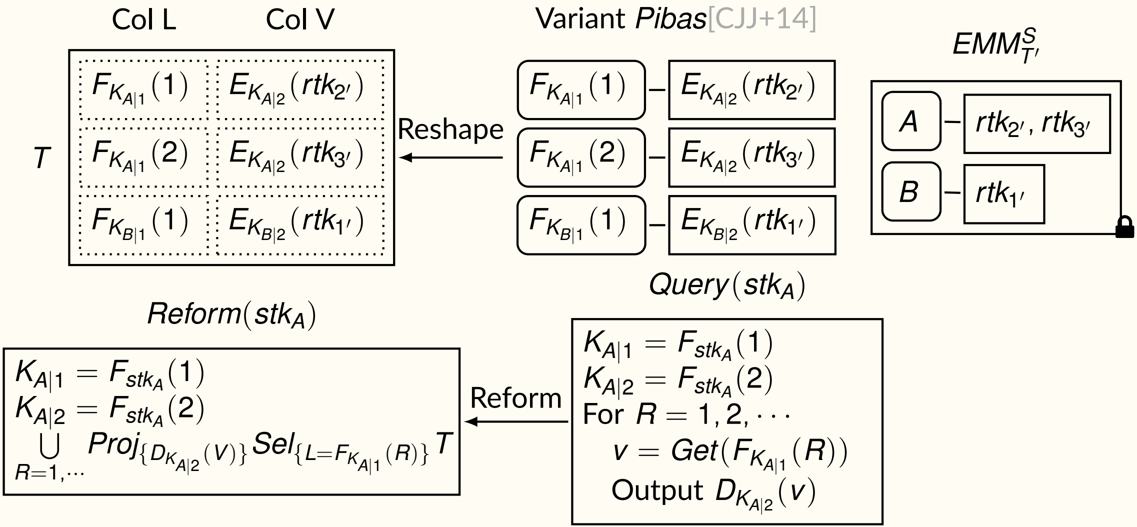
B — $rtk_{1'}$

$Query(stk_A)$

```
$K_{A|1} = F_{stk_A}(1)$
$K_{A|2} = F_{stk_A}(2)$
For $R = 1, 2, \cdots$
    $v = Get(F_{K_{A|1}}(R))$
    Output $D_{K_{A|2}}(v)$
```

- Recursive Common Table Expression in PostgreSQL
- New optimization rule (deferred)

Emulation of Pibas EMM [CJJ+14]



	Col L	Col V
T	$F_{K_{A\|1}}(1)$	$E_{K_{A\|2}}(rtk_{2'})$
	$F_{K_{A\|1}}(2)$	$E_{K_{A\|2}}(rtk_{3'})$
	$F_{K_{B\|1}}(1)$	$E_{K_{B\|2}}(rtk_{1'})$

Reshape

Variant *Pibas*[CJJ+14]

$F_{K_{A\|1}}(1)$	$E_{K_{A\|2}}(rtk_{2'})$
$F_{K_{A\|1}}(2)$	$E_{K_{A\|2}}(rtk_{3'})$
$F_{K_{B\|1}}(1)$	$E_{K_{B\|2}}(rtk_{1'})$

$EMM^{S}_{T'}$

A	$rtk_{2'}, rtk_{3'}$
B	$rtk_{1'}$

$Query(stk_A)$

```
$K_{A|1} = F_{stk_A}(1)$
$K_{A|2} = F_{stk_A}(2)$
For $R = 1, 2, \cdots$
    $v = Get(F_{K_{A|1}}(R))$
    Output $D_{K_{A|2}}(v)$
```

Reform

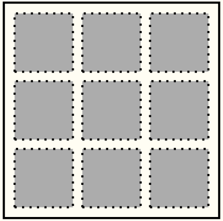
$Reform(stk_A)$

$$K_{A|1} = F_{stk_A}(1)$$
$$K_{A|2} = F_{stk_A}(2)$$
$$\bigcup_{R=1,\cdots} Proj_{\{D_{K_{A|2}}(V)\}} Sel_{\{L=F_{K_{A|1}}(R)\}}\, T$$

Encrypted Conjunction

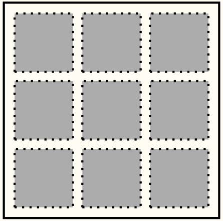
Encrypted Conjunction

Name	Pay	Nation
Alice_1	VISA_1	US_1
Bob_1	PayPal_1	US_2
Bob_2	VISA_2	CAN_1



Encrypted Conjunction

Name	Pay	Nation
Alice$_1$	VISA$_1$	US$_1$
Bob$_1$	PayPal$_1$	US$_2$
Bob$_2$	VISA$_2$	CAN$_1$

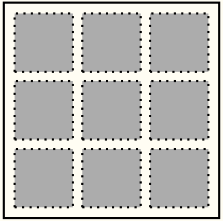


Conjunction

$\sigma_{Play=VISA \wedge Name=Bob}$

Encrypted Conjunction

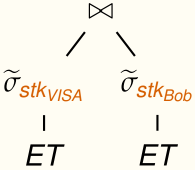
Name	Pay	Nation
Alice_1	VISA_1	US_1
Bob_1	PayPal_1	US_2
Bob_2	VISA_2	CAN_1



Conjunction

$$\sigma_{Play=VISA \wedge Name=Bob}$$

Approach 1



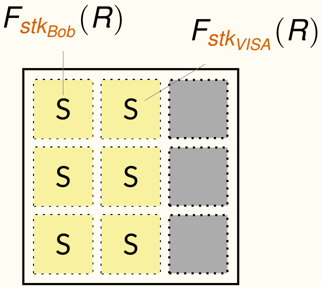
$$\bowtie$$

$\widetilde{\sigma}_{stk_{VISA}}$ $\widetilde{\sigma}_{stk_{Bob}}$

ET ET

Encrypted Conjunction

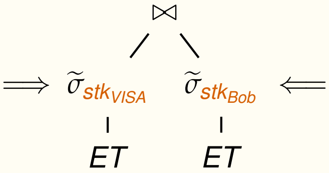
Name	Pay	Nation
Alice_1	VISA_1	US_1
Bob_1	PayPal_1	US_2
Bob_2	VISA_2	CAN_1



Conjunction

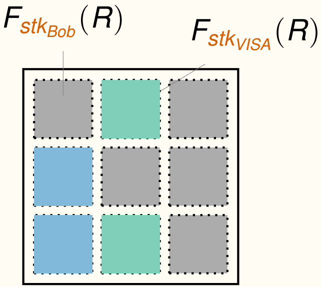
$\sigma_{Play=VISA \wedge Name=Bob}$

Approach 1


$\bowtie$
$\Longrightarrow$
$\widetilde{\sigma}_{stk_{VISA}}$
$\widetilde{\sigma}_{stk_{Bob}}$
$\Longleftarrow$
ET
ET

Encrypted Conjunction

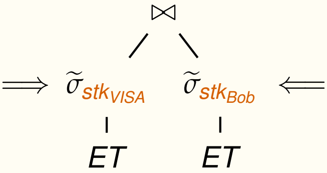
Name	Pay	Nation
Alice_1	VISA_1	US_1
Bob_1	PayPal_1	US_2
Bob_2	VISA_2	CAN_1



Conjunction

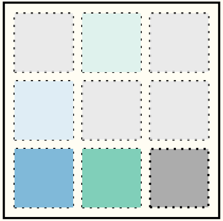
$\sigma_{Play=VISA \wedge Name=Bob}$

Approach 1


$\bowtie$
$\Longrightarrow$
$\widetilde{\sigma}_{stk_{VISA}}$
$\widetilde{\sigma}_{stk_{Bob}}$
$\Longleftarrow$
ET
ET

Encrypted Conjunction

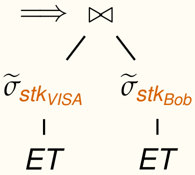
Name	Pay	Nation
Alice_1	VISA_1	US_1
Bob_1	PayPal_1	US_2
Bob_2	VISA_2	CAN_1



Conjunction

$\sigma_{Play=VISA \wedge Name=Bob}$

Approach 1



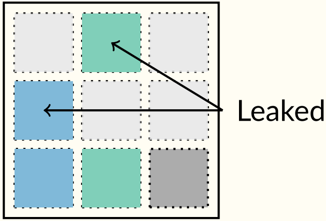
⟹ ⋈
$\widetilde{\sigma}_{stk_{VISA}}$
$\widetilde{\sigma}_{stk_{Bob}}$
ET
ET

Encrypted Conjunction

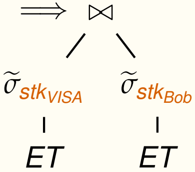
Name	Pay	Nation
Alice$_1$	VISA$_1$	US$_1$
Bob$_1$	PayPal$_1$	US$_2$
Bob$_2$	VISA$_2$	CAN$_1$

Conjunction

$$\sigma_{Play=VISA \wedge Name=Bob}$$

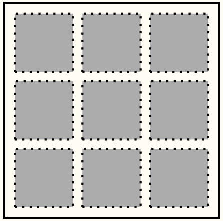


Approach 1

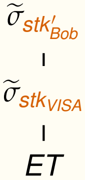

$\Longrightarrow$
$\bowtie$
$\widetilde{\sigma}_{stk_{VISA}}$
$\widetilde{\sigma}_{stk_{Bob}}$
ET
ET

Encrypted Conjunction

Name	Pay	Nation
Alice$_1$	VISA$_1$	US$_1$
Bob$_1$	PayPal$_1$	US$_2$
Bob$_2$	VISA$_2$	CAN$_1$



Approach 2



$\widetilde{\sigma}_{stk'_{Bob}}$

$\widetilde{\sigma}_{stk_{VISA}}$

ET

Encrypted Conjunction

Name	Pay	Nation
Alice_1	VISA_1	US_1
Bob_1	PayPal_1	US_2
Bob_2	VISA_2	CAN_1

▨	S	▨
▨	S	▨
▨	S	▨

Approach 2

$$\widetilde{\sigma}_{stk'_{Bob}}$$

$$|$$

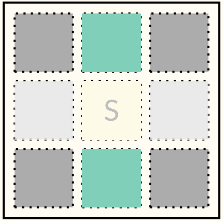
$$\Longrightarrow \widetilde{\sigma}_{stk_{VISA}}$$

$$|$$

$$ET$$

Encrypted Conjunction

Name	Pay	Nation
Alice_1	VISA_1	US_1
Bob_1	PayPal_1	US_2
Bob_2	VISA_2	CAN_1



Approach 2

$\widetilde{\sigma}_{stk'_{Bob}}$

|

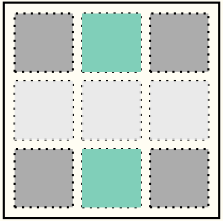
$\Longrightarrow \widetilde{\sigma}_{stk_{VISA}}$

|

ET

Encrypted Conjunction

Name	Pay	Nation
Alice_1	VISA_1	US_1
Bob_1	PayPal_1	US_2
Bob_2	VISA_2	CAN_1



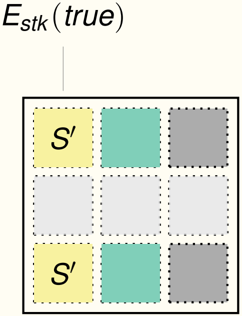
Approach 2

$$\Longrightarrow \widetilde{\sigma}_{stk'_{Bob}} \text{ —— } F_K(Name \,\|\, \boxed{\text{Bob}} \,\|\, Pay \,\|\, \boxed{\text{VISA}})$$

$$\begin{tikzcd} \widetilde{\sigma}_{stk'_{Bob}} \arrow[d, dash] \\ \widetilde{\sigma}_{stk_{VISA}} \arrow[d, dash] \\ ET \end{tikzcd}$$

Encrypted Conjunction

Name	Pay	Nation
$Alice_1$	$VISA_1$	US_1
Bob_1	$PayPal_1$	US_2
Bob_2	$VISA_2$	CAN_1



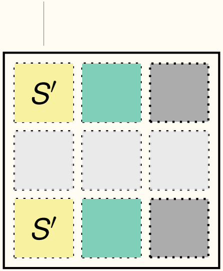
Approach 2

$$\Longrightarrow \begin{tikzcd}
\widetilde{\sigma}_{stk'_{Bob}} \arrow[r, dash] \arrow[d, dash] & F_K(Name \,\|\, \boxed{Bob} \,\|\, Pay \,\|\, \boxed{VISA}) \\
\widetilde{\sigma}_{stk_{VISA}} \arrow[d, dash] & \\
ET &
\end{tikzcd}$$

Encrypted Conjunction

$D_{stk'_{Bob}}(S') = true?$

Name	Pay	Nation
Alice$_1$	VISA$_1$	US$_1$
Bob$_1$	PayPal$_1$	US$_2$
Bob$_2$	VISA$_2$	CAN$_1$



Approach 2

$$\Longrightarrow \widetilde{\sigma}_{stk'_{Bob}} \;—\; F_K(Name \,\|\, \boxed{Bob} \,\|\, Pay \,\|\, \boxed{VISA})$$

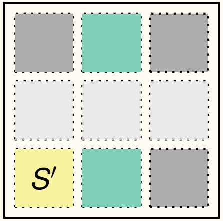
$$\widetilde{\sigma}_{stk_{VISA}}$$

$$ET$$

Encrypted Conjunction

Name	Pay	Nation
Alice$_1$	VISA$_1$	US$_1$
Bob$_1$	PayPal$_1$	US$_2$
$\boxed{\text{Bob}_2}$	$\boxed{\text{VISA}_2}$	CAN$_1$

$$D_{stk'_{Bob}}(S') = true?$$



Approach 2

$$\Longrightarrow \widetilde{\sigma}_{stk'_{Bob}} \text{ — } F_K(Name \| \boxed{\text{Bob}} \| Pay \| \boxed{\text{VISA}})$$

$$|$$

$$\widetilde{\sigma}_{stk_{VISA}}$$

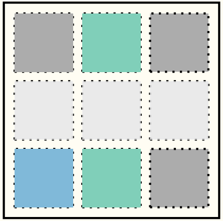
$$|$$

$$ET$$

Encrypted Conjunction

Name	Pay	Nation
Alice$_1$	VISA$_1$	US$_1$
Bob$_1$	PayPal$_1$	US$_2$
Bob$_2$	VISA$_2$	CAN$_1$

$$D_{stk'_{Bob}}(S') = true?$$



Approach 2

$$\Longrightarrow \widetilde{\sigma}_{stk'_{Bob}} \text{ --- } F_K(Name \,\|\, \boxed{Bob} \,\|\, Pay \,\|\, \boxed{VISA})$$

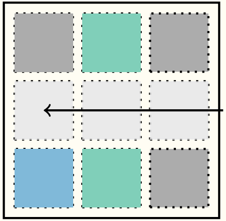
$$\widetilde{\sigma}_{stk_{VISA}}$$

$$ET$$

Encrypted Conjunction

$D_{stk'_{Bob}}(S') = true?$

Name	Pay	Nation
Alice$_1$	VISA$_1$	US$_1$
Bob$_1$	PayPal$_1$	US$_2$
Bob$_2$	VISA$_2$	CAN$_1$



Not leaked

Approach 2

$$\Longrightarrow \widetilde{\sigma}_{stk'_{Bob}} \text{ --- } F_K(Name \parallel \boxed{Bob} \parallel Pay \parallel \boxed{VISA})$$

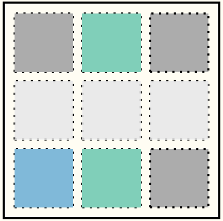
$$\widetilde{\sigma}_{stk_{VISA}}$$

$$ET$$

Encrypted Conjunction

Name	Pay	Nation
Alice_1	VISA_1	US_1
Bob_1	PayPal_1	US_2
Bob_2	VISA_2	CAN_1

Only check on smaller ET



Approach 2

$$\Longrightarrow \widetilde{\sigma}_{stk'_{Bob}} \text{ — } F_K(Name \parallel \boxed{Bob} \parallel Pay \parallel \boxed{VISA})$$

$$|$$

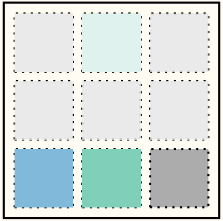
$$\widetilde{\sigma}_{stk_{VISA}}$$

$$|$$

$$ET$$

Encrypted Conjunction

Name	Pay	Nation
Alice_1	VISA_1	US_1
Bob_1	PayPal_1	US_2
Bob_2	VISA_2	CAN_1



Approach 2

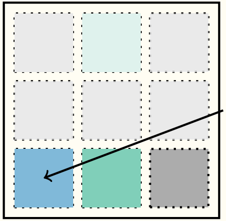
$$\Longrightarrow \widetilde{\sigma}_{stk'_{Bob}} \text{ — } F_K(Name \,\|\, \boxed{\text{Bob}} \,\|\, Pay \,\|\, \boxed{\text{VISA}})$$

$$\widetilde{\sigma}_{stk_{VISA}}$$

$$ET$$

Encrypted Conjunction

Name	Pay	Nation
Alice_1	VISA_1	US_1
Bob_1	PayPal_1	US_2
Bob_2	VISA_2	CAN_1



Leaked only $\bigcap$

Approach 2

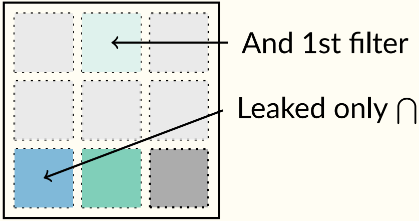
$$\Longrightarrow \widetilde{\sigma}_{stk'_{Bob}} \text{ —— } F_K(Name \,\|\, \boxed{Bob} \,\|\, Pay \,\|\, \boxed{VISA})$$

$$\widetilde{\sigma}_{stk_{VISA}}$$

$$ET$$

Encrypted Conjunction

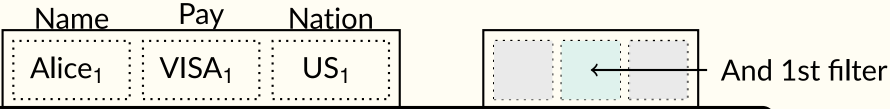
Name	Pay	Nation
Alice_1	VISA_1	US_1
Bob_1	PayPal_1	US_2
Bob_2	VISA_2	CAN_1



Approach 2

$$\begin{tikzcd} \Longrightarrow \widetilde{\sigma}_{stk'_{Bob}} \arrow[r, dash] \arrow[d, dash] & F_K(Name \parallel \boxed{Bob} \parallel Pay \parallel \boxed{VISA}) \\ \widetilde{\sigma}_{stk_{VISA}} \arrow[d, dash] & \\ ET & \end{tikzcd}$$

Encrypted Conjunction



ed only $\bigcap$

- Composition $\Longrightarrow$ leak less
- Ordering $\Longrightarrow$ smaller intermediate ETs $\Longrightarrow$ faster

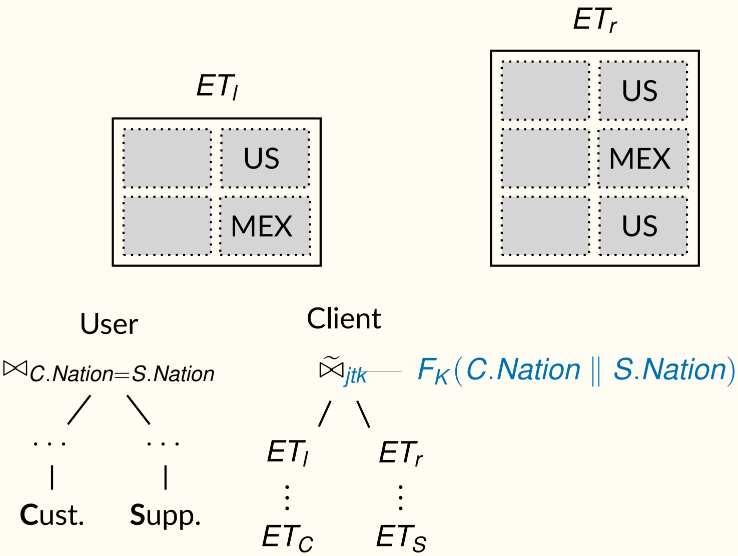
$$\Longrightarrow \widetilde{\sigma}_{stk'_{Bob}} \text{ — } F_K(Name \,\|\, \boxed{\text{Bob}} \,\|\, Pay \,\|\, \boxed{\text{VISA}})$$

$\widetilde{\sigma}_{stk_{VISA}}$

ET

Reducing Leakage

Reducing Leakage



ET_r

	US
	MEX
	US

ET_l

	US
	MEX

User

$\bowtie_{C.Nation=S.Nation}$

Cust. Supp.

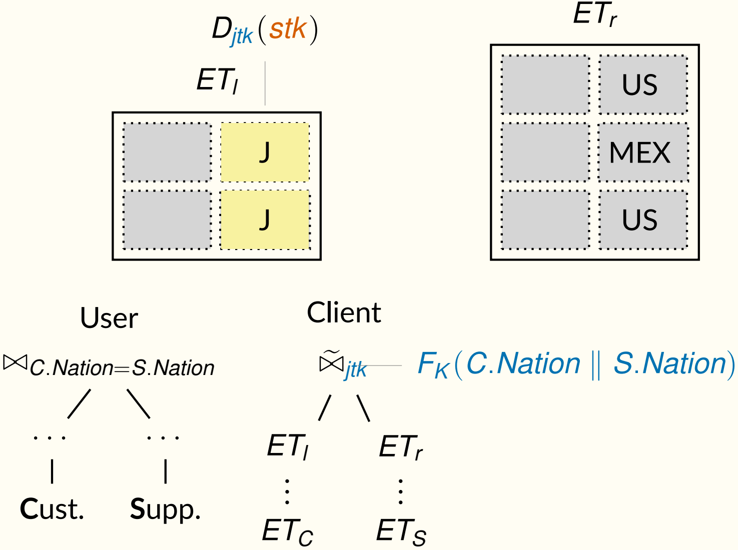
Client

$\widetilde{\bowtie}_{jtk}$ — $F_K(C.Nation \,\|\, S.Nation)$

ET_l ET_r

ET_C ET_S

Reducing Leakage



$D_{jtk}(stk)$

ET_l

	J
	J

ET_r

	US
	MEX
	US

User

$\bowtie_{C.Nation=S.Nation}$

$\cdots$ $\cdots$

Cust. Supp.

Client

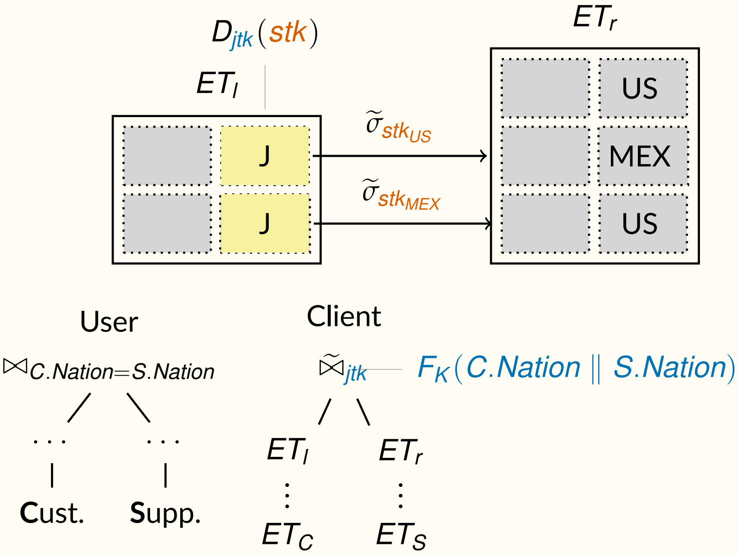
$\widetilde{\bowtie}_{jtk}$ $F_K(C.Nation \parallel S.Nation)$

ET_l ET_r

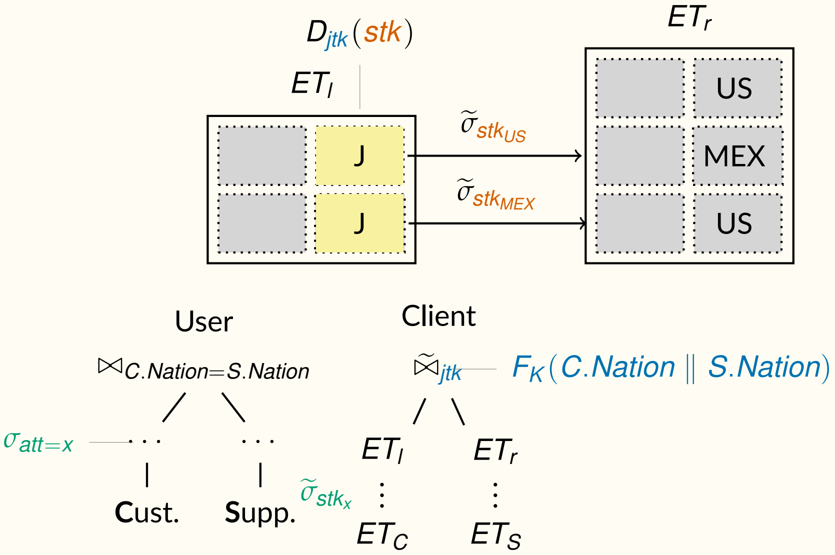
$\vdots$ $\vdots$

ET_C ET_S

Reducing Leakage


$D_{jtk}(stk)$
ET_l
J
J
$\widetilde{\sigma}_{stk_{US}}$
$\widetilde{\sigma}_{stk_{MEX}}$
ET_r
US
MEX
US
User
$\bowtie_{C.Nation=S.Nation}$
. . .
. . .
Cust.
Supp.
Client
$\widetilde{\bowtie}_{jtk}$
$F_K(C.Nation \parallel S.Nation)$
ET_l
ET_r
ET_C
ET_S

Reducing Leakage



$D_{jtk}(stk)$

ET_l

J

J

$\widetilde{\sigma}_{stk_{US}}$

$\widetilde{\sigma}_{stk_{MEX}}$

ET_r

US

MEX

US

User

$\bowtie_{C.Nation=S.Nation}$

$\sigma_{att=x}$ $\cdots$ $\cdots$

Cust. Supp.

Client

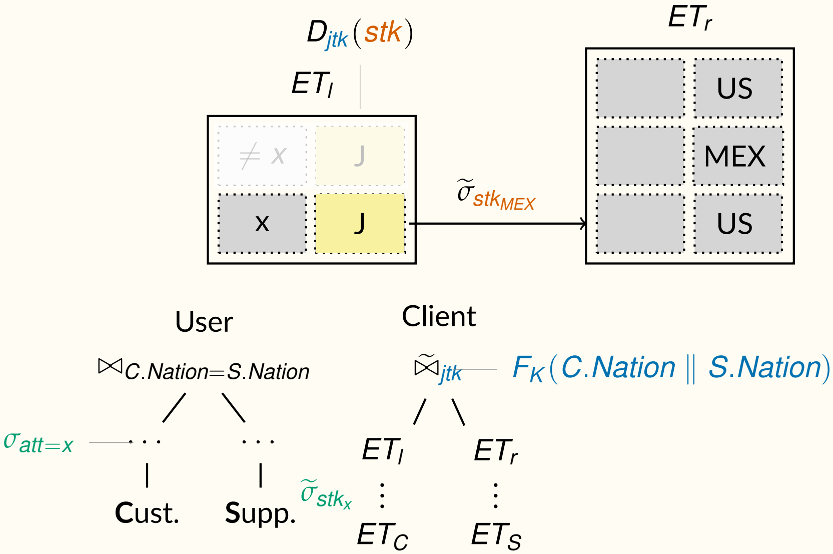
$\widetilde{\bowtie}_{jtk}$ $F_K(C.Nation \parallel S.Nation)$

ET_l ET_r

$\widetilde{\sigma}_{stk_x}$ $\vdots$ $\vdots$

ET_C ET_S

Reducing Leakage



$D_{jtk}(stk)$

ET_l

$\neq x$ | J

x | J

$\widetilde{\sigma}_{stk_{MEX}}$

ET_r

US

MEX

US

User

$\bowtie_{C.Nation=S.Nation}$

$\sigma_{att=x}$

Cust.

Supp.

Client

$\widetilde{\bowtie}_{jtk}$

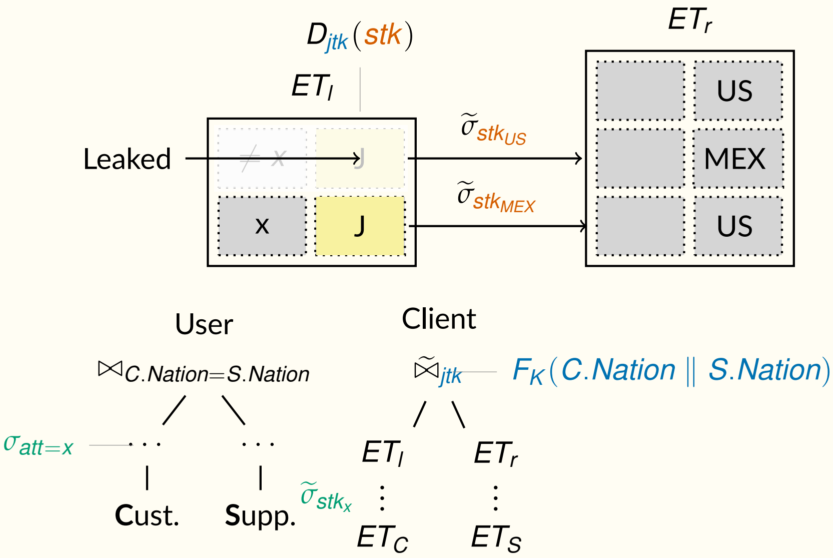
$F_K(C.Nation \parallel S.Nation)$

ET_l ET_r

$\widetilde{\sigma}_{stk_x}$

ET_C ET_S

Reducing Leakage



$D_{jtk}(stk)$

ET_l

ET_r

Leaked

$\neq x$ J

x J

$\widetilde{\sigma}_{stk_{US}}$

$\widetilde{\sigma}_{stk_{MEX}}$

US

MEX

US

User

Client

$\bowtie_{C.Nation=S.Nation}$

$\sigma_{att=x}$ $\cdots$ $\cdots$

Cust. Supp.

$\widetilde{\bowtie}_{jtk}$ $F_K(C.Nation \parallel S.Nation)$

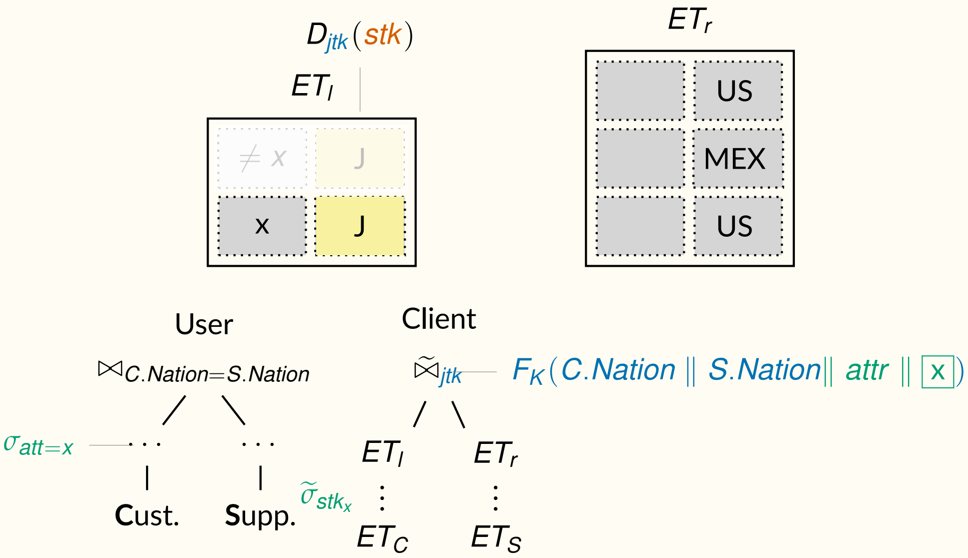
$\widetilde{\sigma}_{stk_x}$

ET_l ET_r

$\vdots$ $\vdots$

ET_C ET_S

Reducing Leakage



$D_{jtk}(stk)$

ET_l

$\neq x$	J
x	J

ET_r

	US
	MEX
	US

User

$\bowtie_{C.Nation=S.Nation}$

$\sigma_{att=x}$

Cust.

Supp.

Client

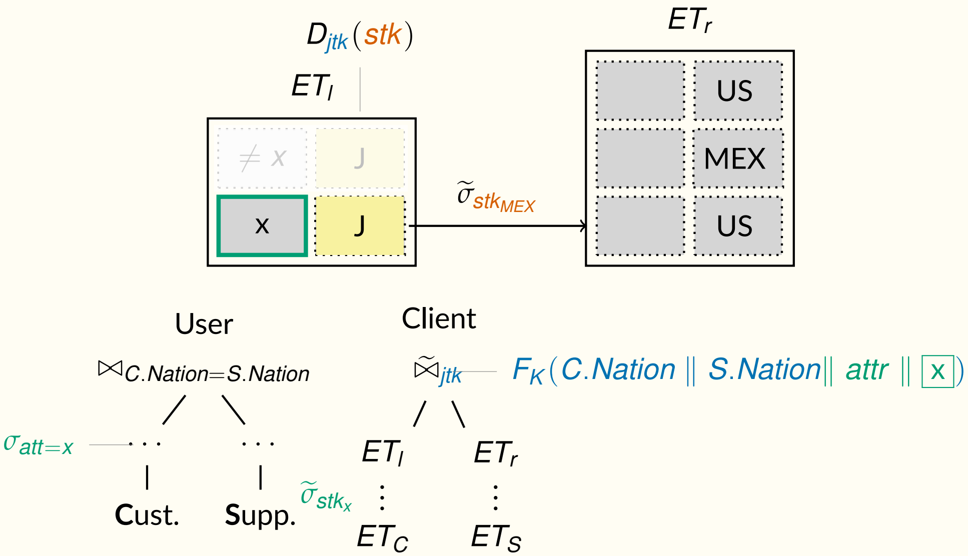
$\widetilde{\bowtie}_{jtk}$ $F_K(C.Nation \,\|\, S.Nation \| \; attr \,\|\, \boxed{x})$

$\widetilde{\sigma}_{stk_x}$

ET_l ET_r

ET_C ET_S

Reducing Leakage



$D_{jtk}(stk)$

ET_l

$\neq x$ | J

x | J

$\widetilde{\sigma}_{stk_{MEX}}$

ET_r

US

MEX

US

User

$\bowtie_{C.Nation=S.Nation}$

$\sigma_{att=x}$ . . .

. . .

Cust.

Supp.

Client

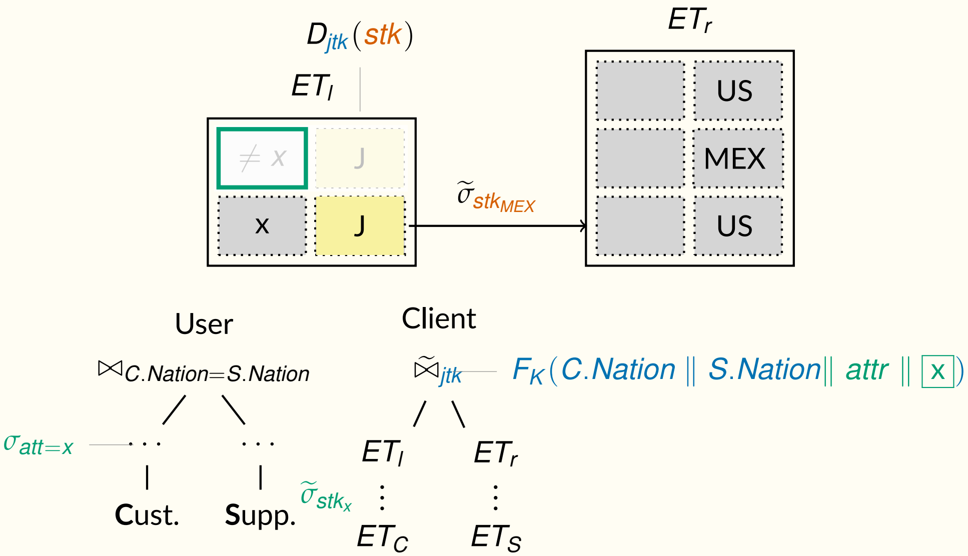
$\widetilde{\bowtie}_{jtk}$ $F_K(C.Nation \parallel S.Nation \parallel attr \parallel \boxed{x})$

$\widetilde{\sigma}_{stk_x}$

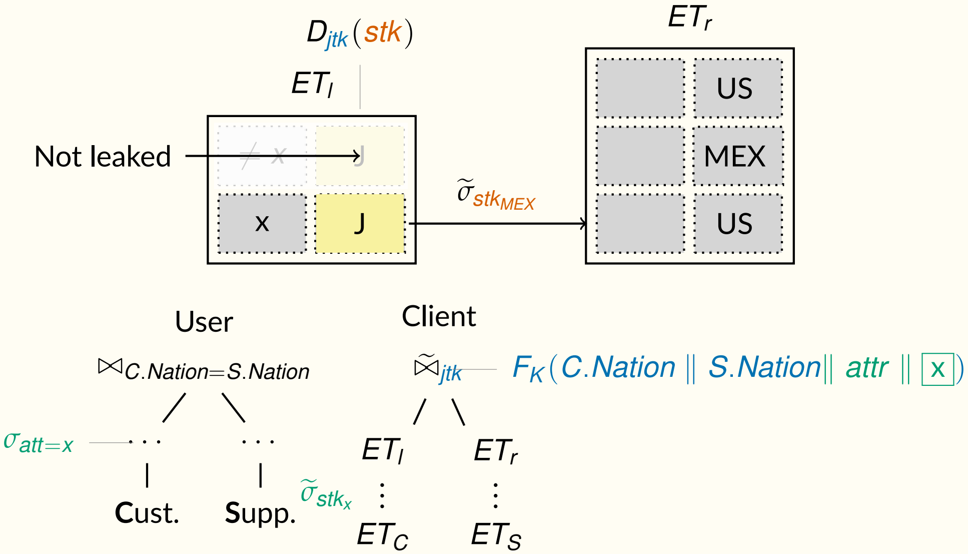
ET_l ET_r

ET_C ET_S

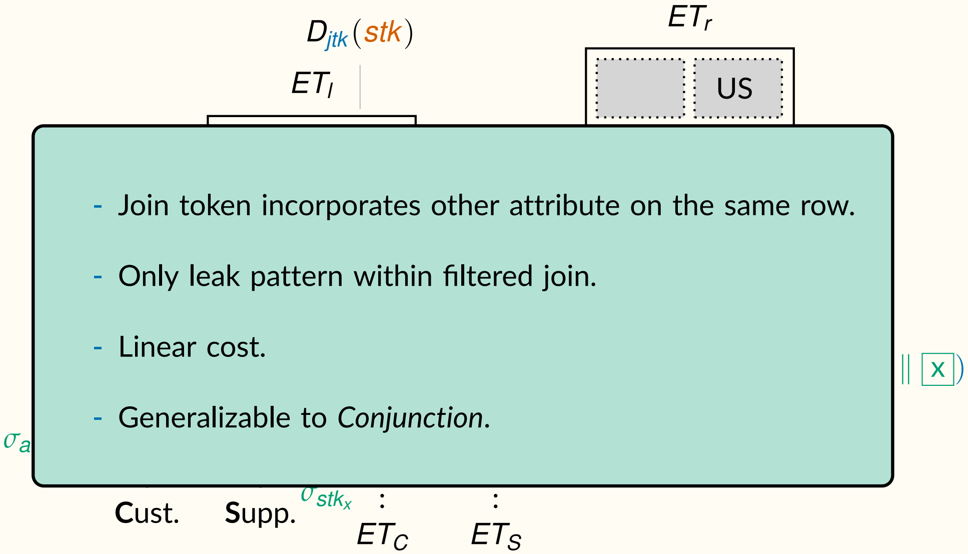
Reducing Leakage


$D_{jtk}(stk)$
ET_l
$\neq x$
J
x
J
$\widetilde{\sigma}_{stk_{MEX}}$
ET_r
US
MEX
US
User
$\bowtie_{C.Nation=S.Nation}$
$\sigma_{att=x}$
Cust.
Supp.
Client
$\widetilde{\bowtie}_{jtk}$
$F_K(C.Nation \parallel S.Nation \parallel attr \parallel \boxed{x})$
ET_l
ET_r
$\widetilde{\sigma}_{stk_x}$
ET_C
ET_S

Reducing Leakage


$D_{jtk}(stk)$
ET_l
ET_r
Not leaked
$\neq x$
J
x
J
$\widetilde{\sigma}_{stk_{MEX}}$
US
MEX
US
User
Client
$\bowtie_{C.Nation=S.Nation}$
$\sigma_{att=x}$
Cust.
Supp.
$\widetilde{\bowtie}_{jtk}$
$F_K(C.Nation \parallel S.Nation \parallel attr \parallel \boxed{x})$
$\widetilde{\sigma}_{stk_x}$
ET_l
ET_r
ET_C
ET_S

Reducing Leakage

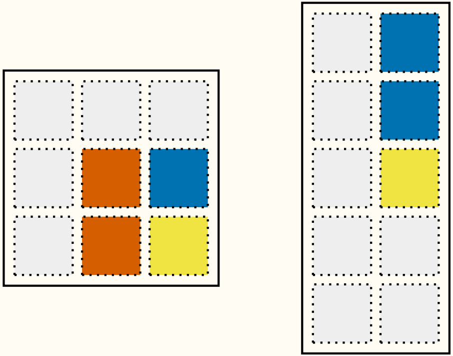


- Join token incorporates other attribute on the same row.
- Only leak pattern within filtered join.
- Linear cost.
- Generalizable to *Conjunction*.

Security

Simulation

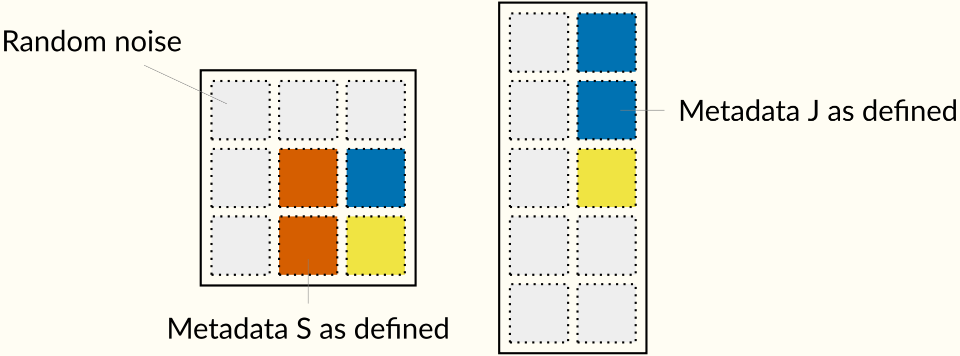
Show: leaks patterns in query result and nothing else.



Simulation

Show: leaks patterns in query result and nothing else.

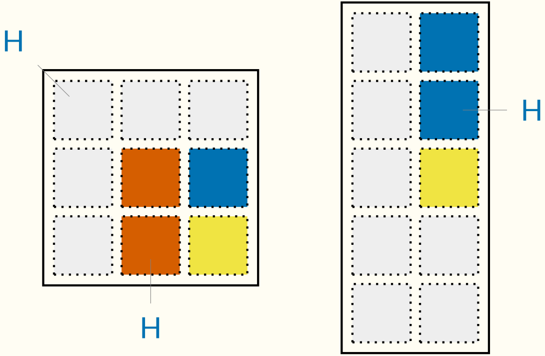
- Non-adaptive security: swap non-queried cells with random noise



Simulation

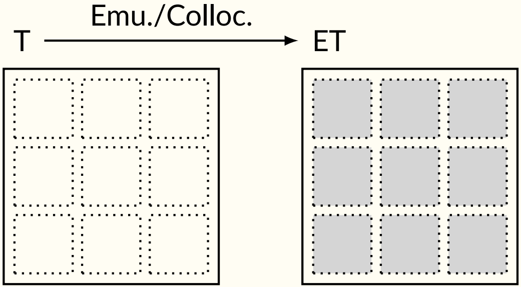
Show: leaks patterns in query result and nothing else.

- Adaptive security in ROM: $F_K(x) \doteq H(K \parallel x)$, $E_K(m) \doteq (r, H(K \parallel r) \oplus m)$ for random K, r.

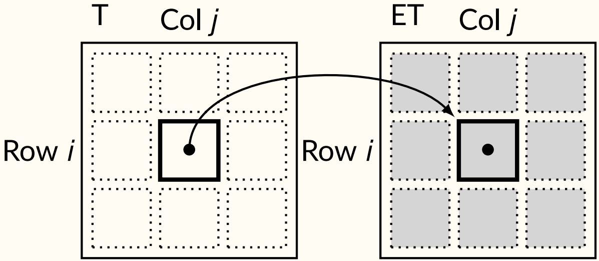


Encrypted Table

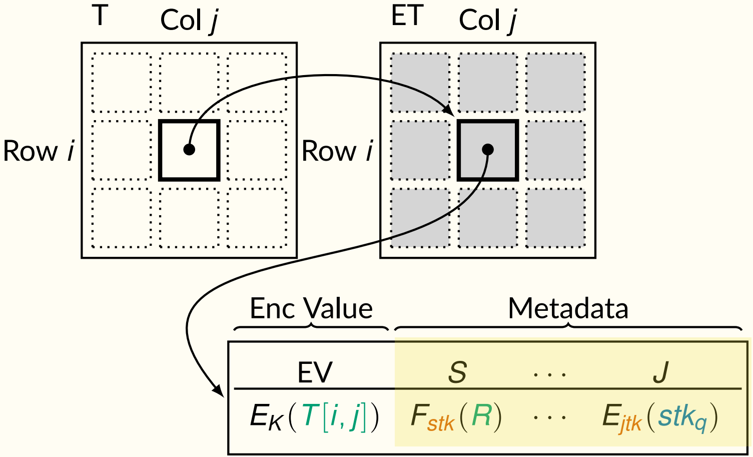
Encrypted Table



Encrypted Table



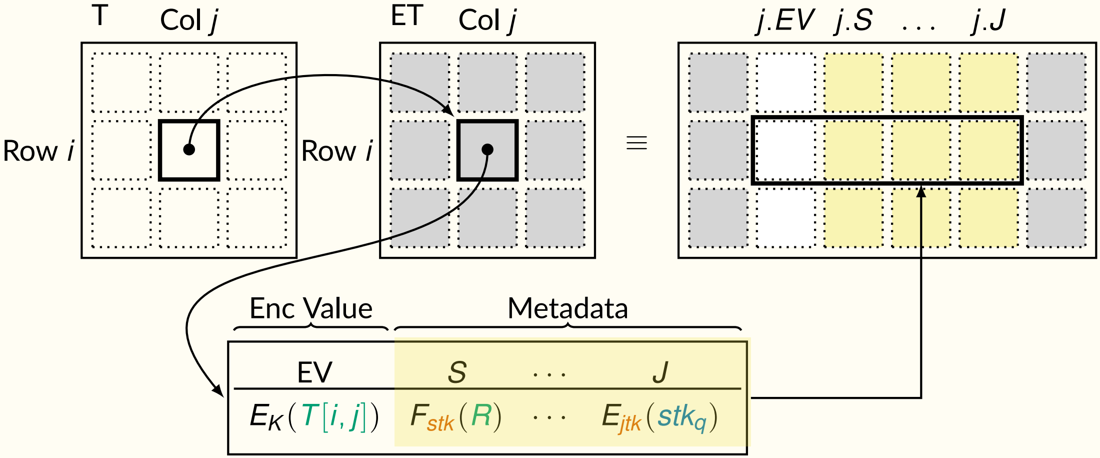
Encrypted Table



EV	S	$\cdots$	J
$E_K(T[i,j])$	$F_{stk}(R)$	$\cdots$	$E_{jtk}(stk_q)$

Collocated from $EMM^R, EMM^S, EMM^J, \cdots$

Encrypted Table



Collocated from $EMM^R, EMM^S, EMM^J, \cdots$

Encrypted Join

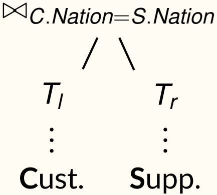
T_l

	US
	MEX

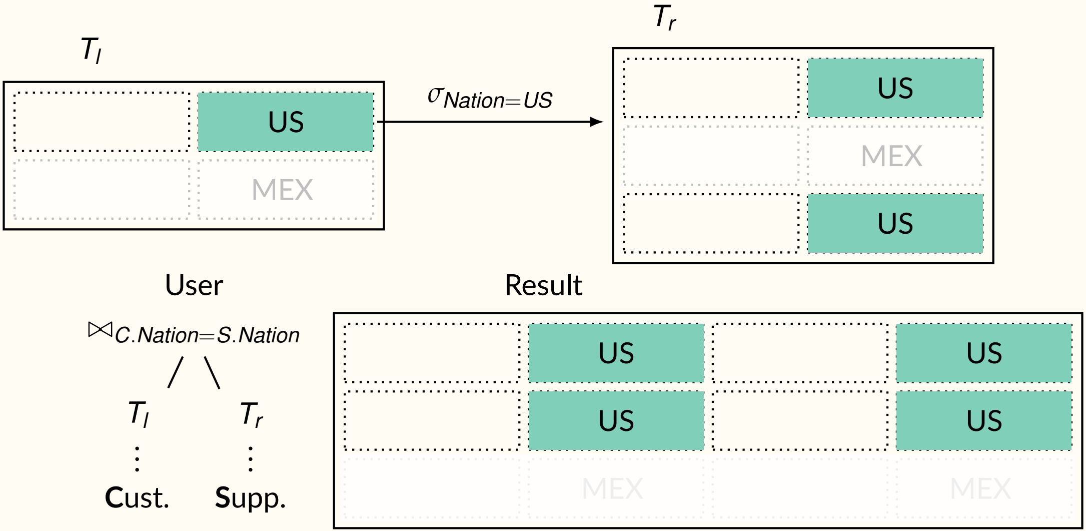
T_r

	US
	MEX
	US

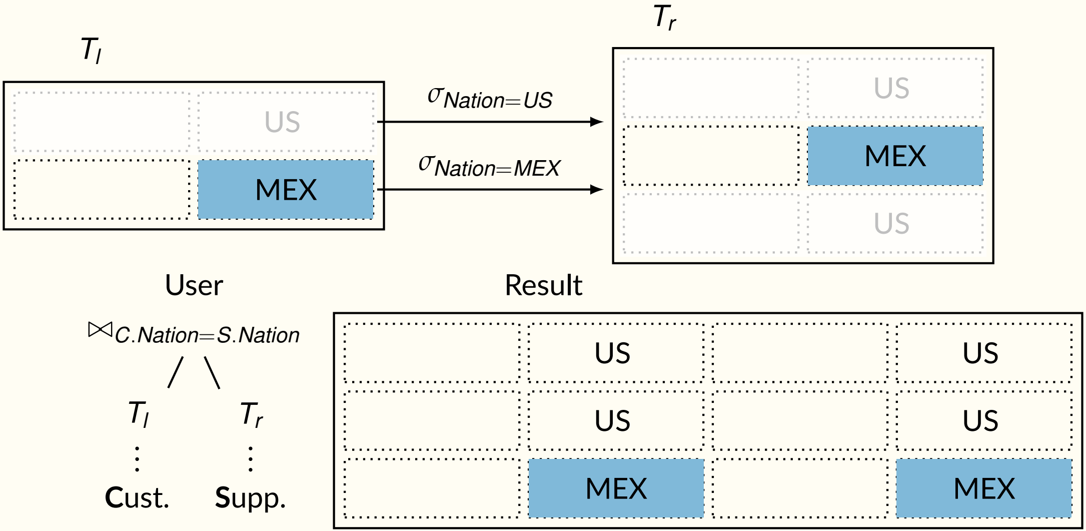
User



Encrypted Join



Encrypted Join



T_l

	US
	MEX

$\sigma_{Nation=US}$

$\sigma_{Nation=MEX}$

T_r

	US
	MEX
	US

User

$\bowtie_{C.Nation=S.Nation}$

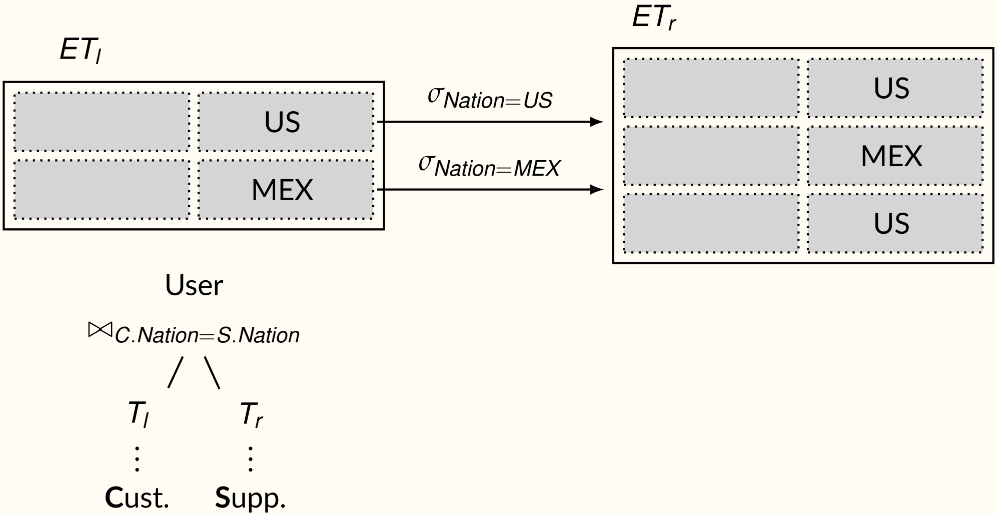
T_l T_r

Cust. Supp.

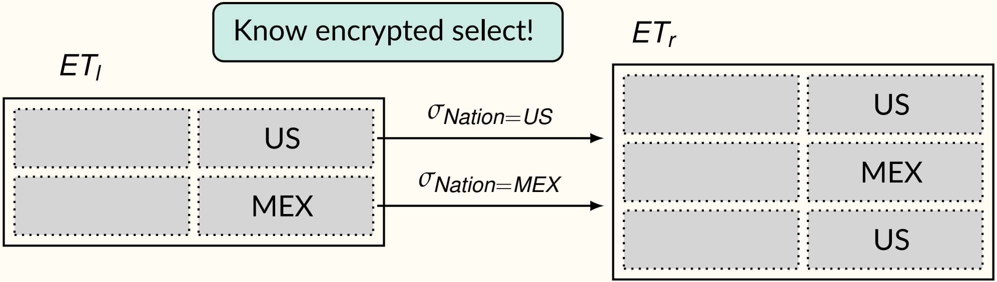
Result

	US		US
	US		US
	MEX		MEX

Encrypted Join



Encrypted Join



Know encrypted select!

ET_l

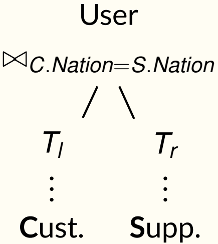
	US
	MEX

$\sigma_{Nation=US}$

$\sigma_{Nation=MEX}$

ET_r

	US
	MEX
	US



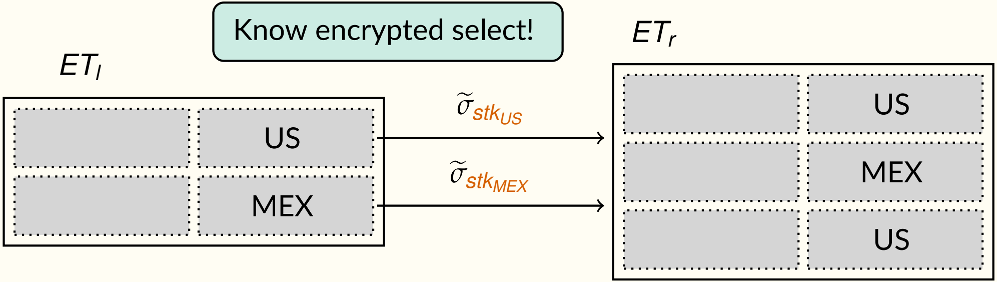
User

$\bowtie_{C.Nation=S.Nation}$

T_l T_r

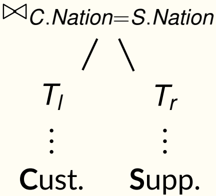
Cust. **Supp.**

Encrypted Join

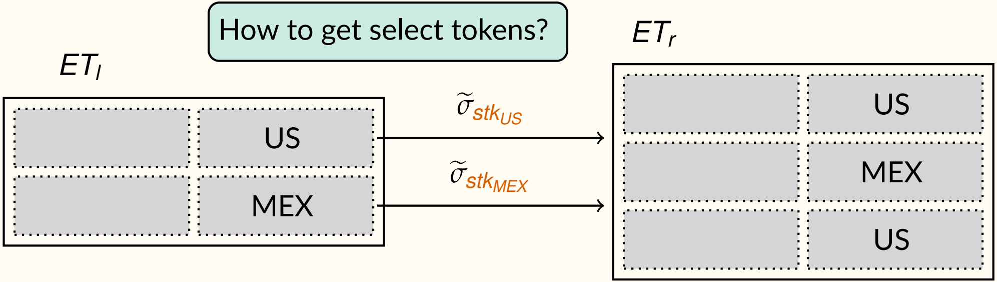

Know encrypted select!

ET_l
US
MEX
$\widetilde{\sigma}_{stk_{US}}$
$\widetilde{\sigma}_{stk_{MEX}}$
ET_r
US
MEX
US

User



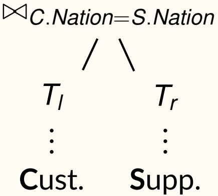
Encrypted Join



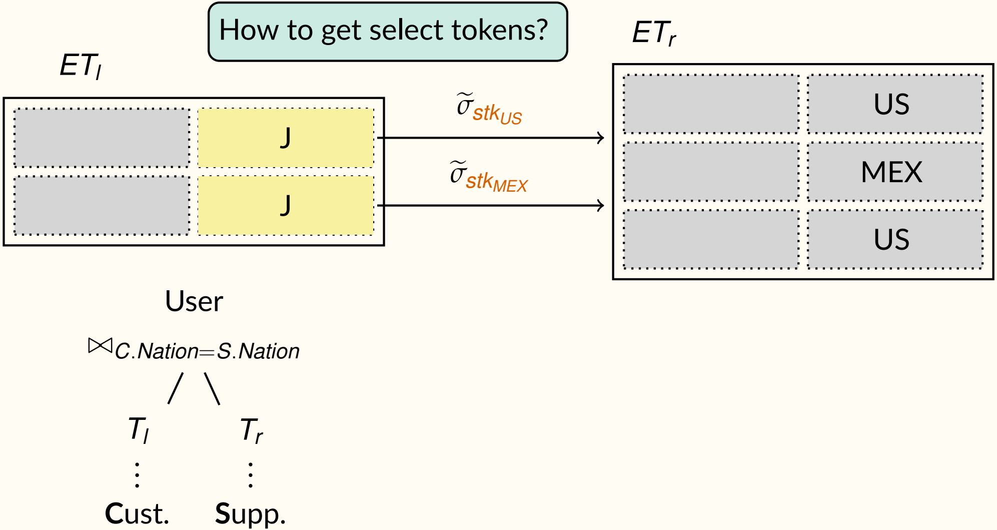
How to get select tokens?

ET_l
US
MEX
$\widetilde{\sigma}_{stk_{US}}$
$\widetilde{\sigma}_{stk_{MEX}}$
ET_r
US
MEX
US

User



Encrypted Join



How to get select tokens?

ET_l

J

J

$\widetilde{\sigma}_{stk_{US}}$

$\widetilde{\sigma}_{stk_{MEX}}$

ET_r

US

MEX

US

User

$\bowtie_{C.Nation=S.Nation}$

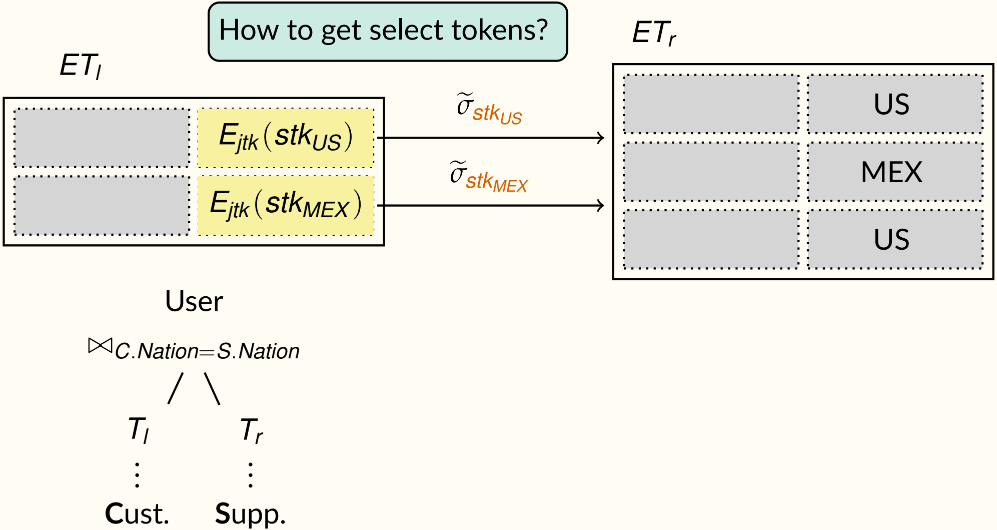
T_l

T_r

Cust.

Supp.

Encrypted Join

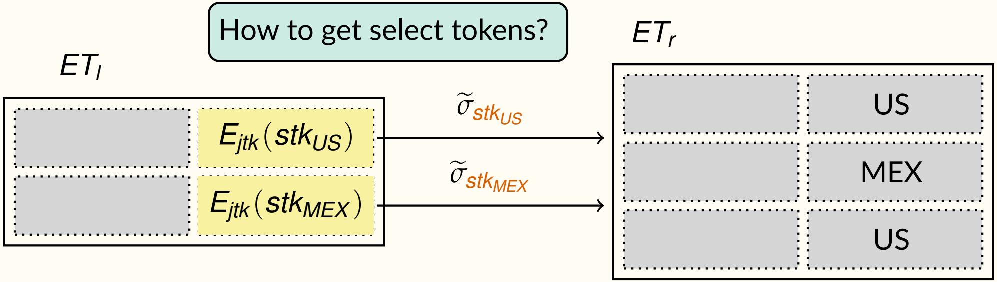


How to get select tokens?

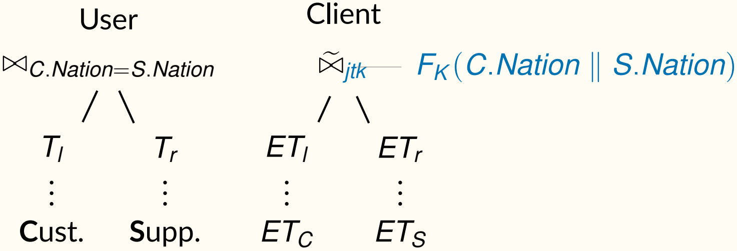
ET_l
$E_{jtk}(stk_{US})$
$E_{jtk}(stk_{MEX})$
$\widetilde{\sigma}_{stk_{US}}$
$\widetilde{\sigma}_{stk_{MEX}}$
ET_r
US
MEX
US
User
$\bowtie_{C.Nation=S.Nation}$
T_l
T_r
Cust.
Supp.

Encrypted Join

How to get select tokens?



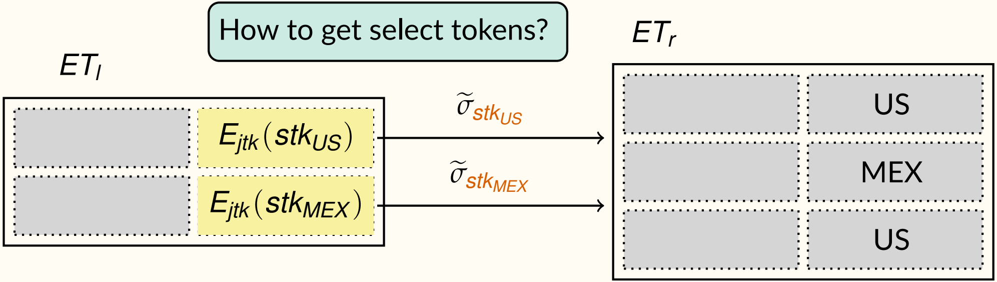
ET_l
$E_{jtk}(stk_{US})$
$E_{jtk}(stk_{MEX})$
$\widetilde{\sigma}_{stk_{US}}$
$\widetilde{\sigma}_{stk_{MEX}}$
ET_r
US
MEX
US

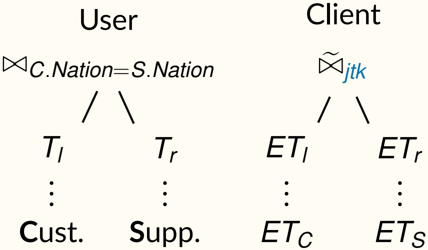


User
$\bowtie_{C.Nation=S.Nation}$
T_l
T_r
Cust.
Supp.
Client
$\widetilde{\bowtie}_{jtk}$
$F_K(C.Nation \parallel S.Nation)$
ET_l
ET_r
ET_C
ET_S

Encrypted Join

How to get select tokens?

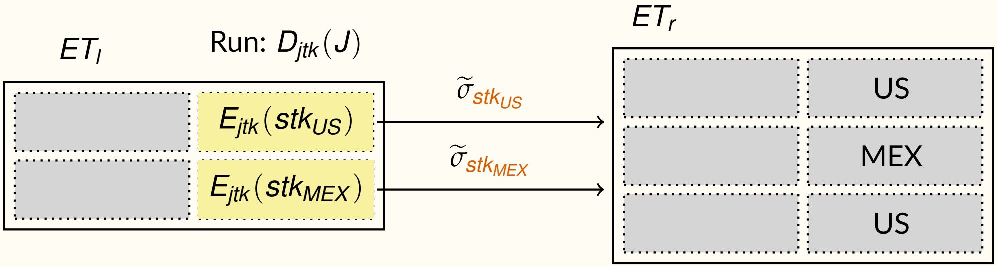

ET_l
$E_{jtk}(stk_{US})$
$E_{jtk}(stk_{MEX})$
$\widetilde{\sigma}_{stk_{US}}$
$\widetilde{\sigma}_{stk_{MEX}}$
ET_r
US
MEX
US

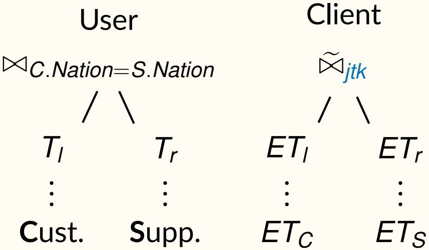


Server

```
For i-th row $\in ET_l$:
    Let $stk_i \leftarrow D_{jtk}(J_i)$,
    Output $ET_l[i,\cdot] \times \widetilde{\sigma}_{stk_i}(ET_r)$
```

Encrypted Join

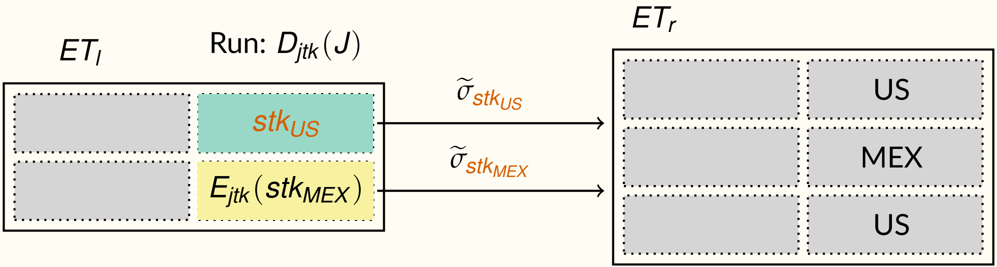


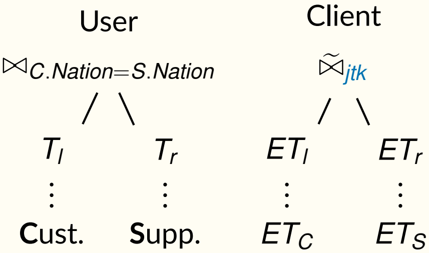


Server

```
For i-th row $\in ET_l$:
    Let $stk_i \leftarrow D_{jtk}(J_i)$,
    Output $ET_l[i,\cdot] \times \tilde{\sigma}_{stk_i}(ET_r)$
```

Encrypted Join

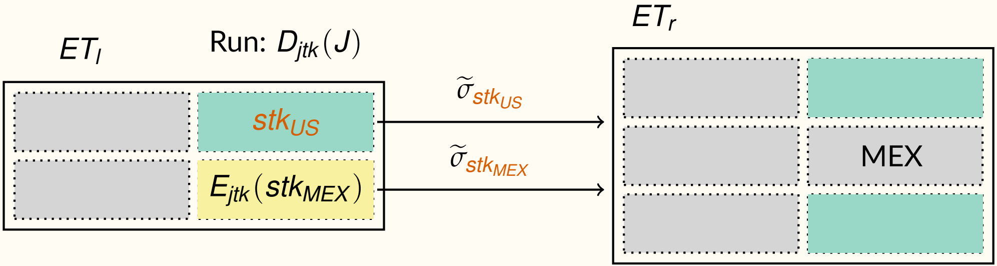


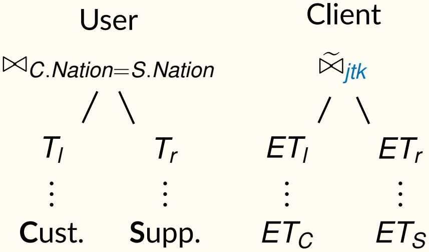


Server

```
For i-th row $\in ET_l$:
    Let $stk_i \leftarrow D_{jtk}(J_i)$,
    Output $ET_l[i,\cdot] \times \tilde{\sigma}_{stk_i}(ET_r)$
```

Encrypted Join

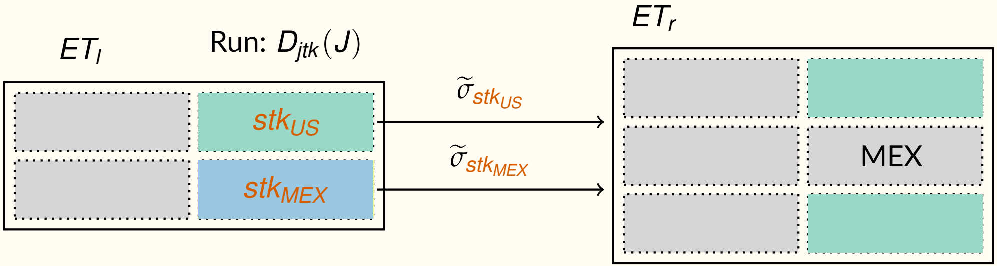

ET_l
Run: $D_{jtk}(J)$
stk_{US}
$E_{jtk}(stk_{MEX})$
$\widetilde{\sigma}_{stk_{US}}$
$\widetilde{\sigma}_{stk_{MEX}}$
ET_r
MEX



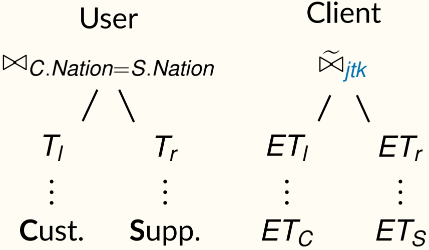
Server

```
For i-th row $\in ET_l$:
    Let $stk_i \leftarrow D_{jtk}(J_i)$,
    Output $ET_l[i, \cdot] \times \widetilde{\sigma}_{stk_i}(ET_r)$
```

Encrypted Join



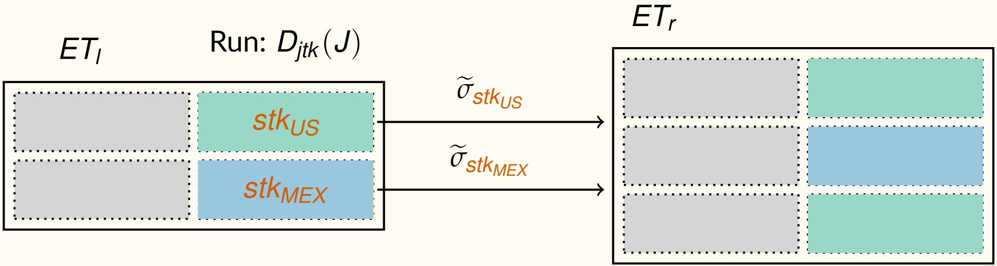
ET_l
Run: $D_{jtk}(J)$
stk_{US}
stk_{MEX}
$\widetilde{\sigma}_{stk_{US}}$
$\widetilde{\sigma}_{stk_{MEX}}$
ET_r
MEX



Server

```
For i-th row $\in ET_l$:
    Let $stk_i \leftarrow D_{jtk}(J_i)$,
    Output $ET_l[i, \cdot] \times \widetilde{\sigma}_{stk_i}(ET_r)$
```

Encrypted Join



ET_l

Run: $D_{jtk}(J)$

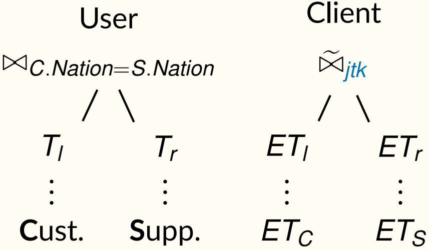
stk_{US}

stk_{MEX}

$\widetilde{\sigma}_{stk_{US}}$

$\widetilde{\sigma}_{stk_{MEX}}$

ET_r



User

$\bowtie_{C.Nation=S.Nation}$

T_l T_r

Cust. Supp.

Client

$\widetilde{\bowtie}_{jtk}$

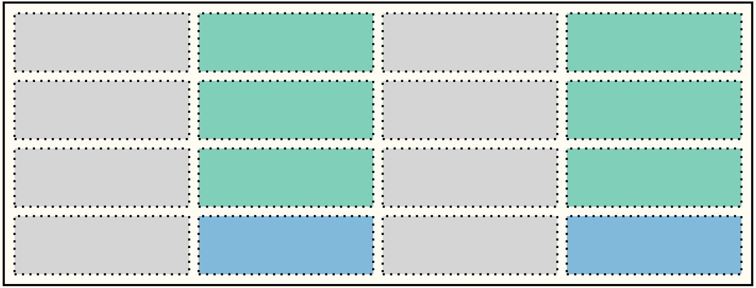
ET_l ET_r

ET_C ET_S

Server

```
For i-th row $\in ET_l$:
    Let $stk_i \leftarrow D_{jtk}(J_i)$,
    Output $ET_l[i,\cdot] \times \widetilde{\sigma}_{stk_i}(ET_r)$
```

Encrypted Join



Encrypted Join

- Linear cost.
- Extension for leakage redunction for filtered joins at linear cost.

Fixed-Point Operator

- For recursion: common in Encryted Selection and Encrypted Join

$$\bigcup_{R \in Z^+} \sigma_{S = F_{stk}(R)} ET$$

Fixed-Point Operator

- For recursion: common in Encryted Selection and Encrypted Join

$$\bigcup_{R \in Z^{+}} \sigma_{S=F_{stk}(R)} ET$$

- Extension beyond relational algebra
  - Not in all database systems: SparkSQL
  - Postgres 8+/MySQL 8+/SQL Server 2005+: Recursive Common Table Expression
  - Oracle 11g Release 2: Recursive Subquery Factoring / CONNECT BY

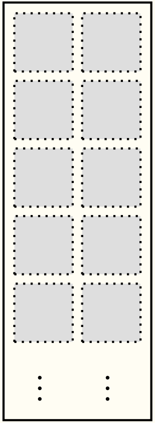
Fixed-Point Operator

- For recursion: common in Encryted Selection and Encrypted Join

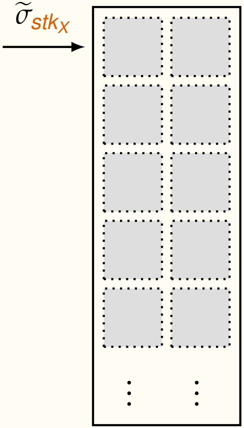
$$\bigcup_{R \in Z^{+}} \sigma_{S=F_{stk}(R)} ET$$

- Extension beyond relational algebra
  - Not in all database systems: SparkSQL
  - Postgres 8+/MySQL 8+/SQL Server 2005+: Recursive Common Table Expression
  - Oracle 11g Release 2: Recursive Subquery Factoring / CONNECT BY
- Important to optimize for efficiency
  - Data Paralllel Rewrite
  - Join Direction Reorder
  - Hash Join Reuse

Fixed-Point Operator

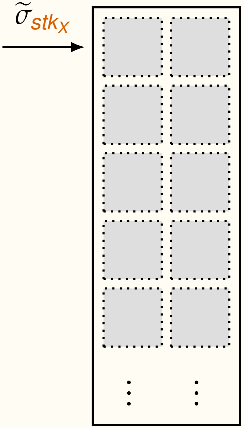


Fixed-Point Operator



Fixed-Point Operator

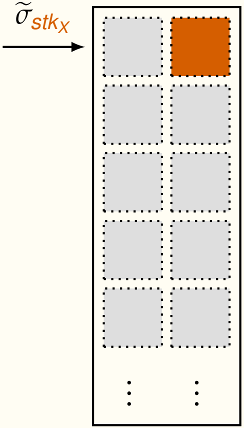
Run: $\bigcup_{R \in Z^+} \sigma_{S=F_{stk}(R)}$

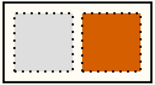


Fixed-Point Operator

Run: $\bigcup_{R \in Z^+} \sigma_{S=F_{stk}(R)}$

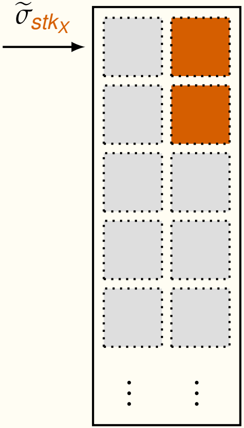
For $R = 1$



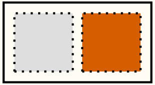


Fixed-Point Operator

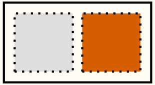
Run: $\bigcup_{R \in Z^+} \sigma_{S=F_{stk}(R)}$



For $R = 1$



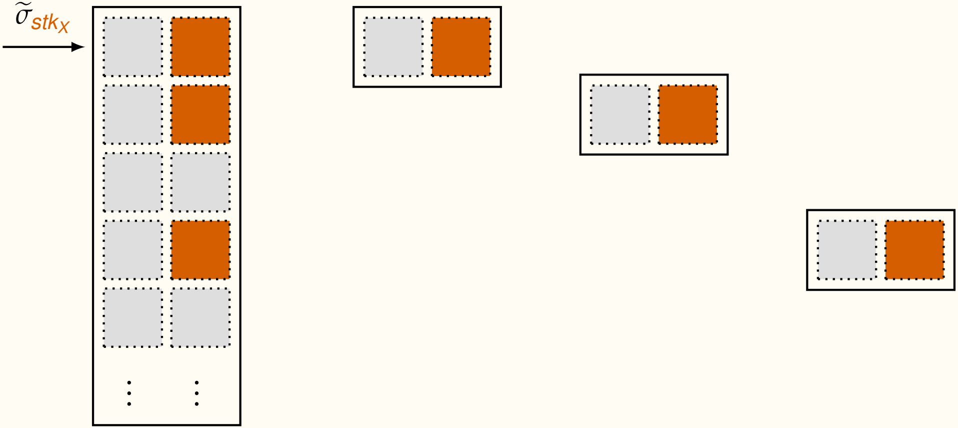
For $R = 2$



Fixed-Point Operator

Run: $\bigcup_{R\in Z^+} \sigma_{S=F_{stk}(R)}$

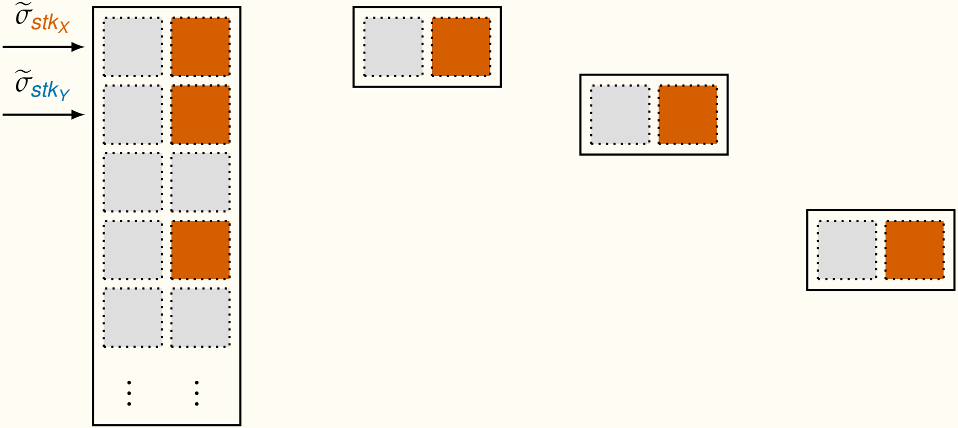
For $R = 1$ For $R = 2$ For $R = 3$



Fixed-Point Operator

Run: $\bigcup_{R\in Z^+} \sigma_{S=F_{stk}(R)}$

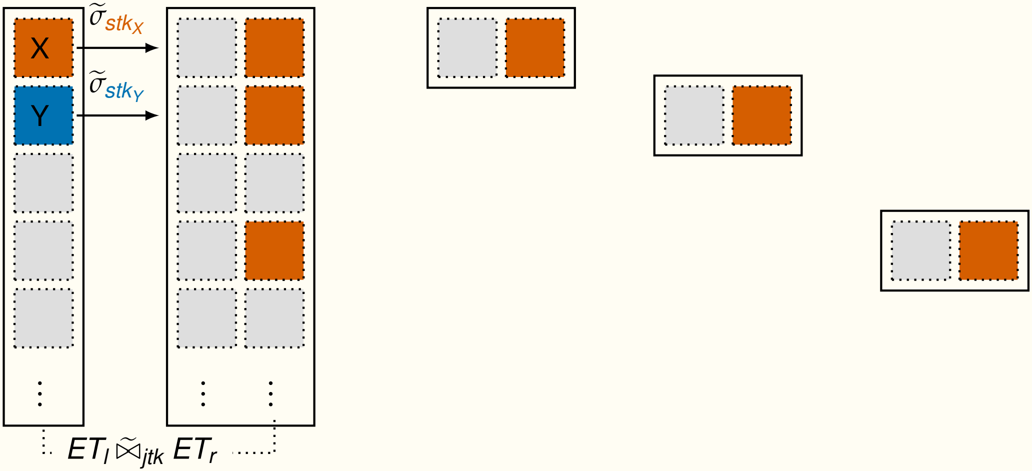
For $R = 1$ For $R = 2$ For $R = 3$



Fixed-Point Operator

Run: $\bigcup_{R \in Z^+} \sigma_{S=F_{stk}(R)}$

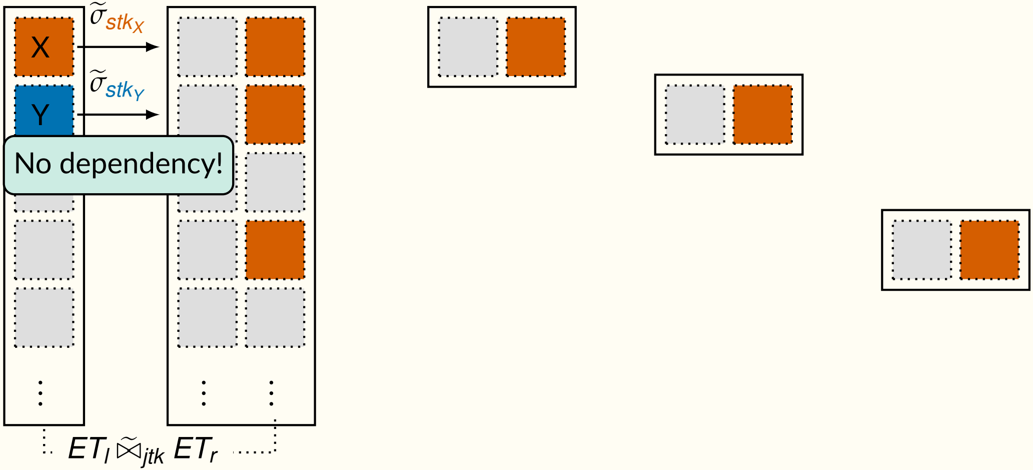
For $R = 1$ For $R = 2$ For $R = 3$


X
Y
$\widetilde{\sigma}_{stk_X}$
$\widetilde{\sigma}_{stk_Y}$
$ET_l \,\widetilde{\bowtie}_{jtk}\, ET_r$

Fixed-Point Operator

Run: $\bigcup_{R\in Z^{+}} \sigma_{S=F_{stk}(R)}$

For $R = 1$ For $R = 2$ For $R = 3$

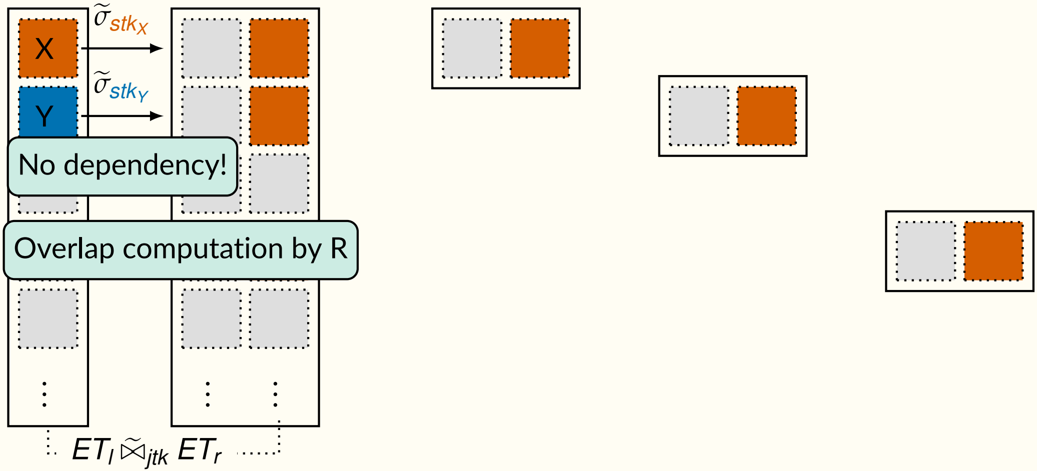

X
Y
$\widetilde{\sigma}_{stk_X}$
$\widetilde{\sigma}_{stk_Y}$
$\vdots$
$ET_l \mathbin{\widetilde{\bowtie}_{jtk}} ET_r$

No dependency!

Fixed-Point Operator

Run: $\bigcup_{R \in Z^{+}} \sigma_{S=F_{stk}(R)}$

For $R = 1$ For $R = 2$ For $R = 3$



X
Y
$\widetilde{\sigma}_{stk_X}$
$\widetilde{\sigma}_{stk_Y}$
$ET_l \widetilde{\bowtie}_{jtk} ET_r$

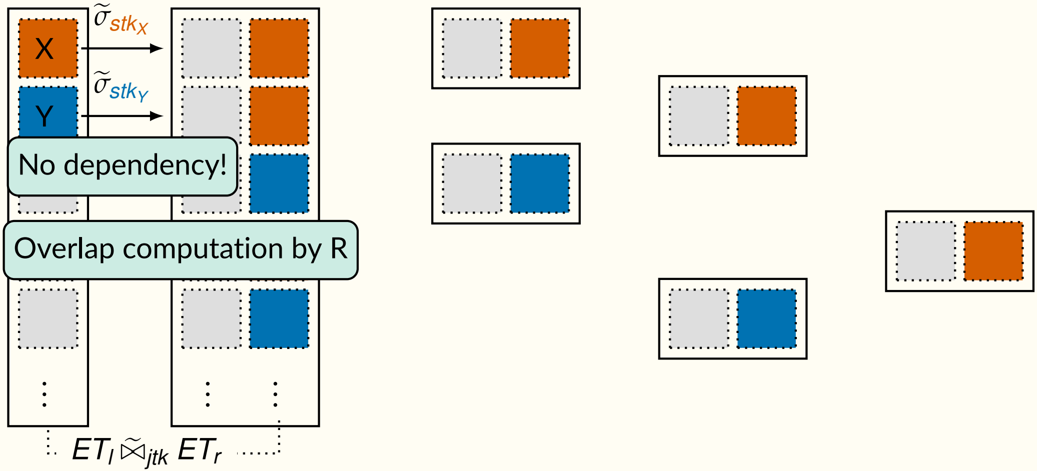
No dependency!

Overlap computation by R

Fixed-Point Operator

Run: $\bigcup_{R \in Z^+} \sigma_{S=F_{stk}(R)}$

For $R = 1$ For $R = 2$ For $R = 3$


X
Y
$\widetilde{\sigma}_{stk_X}$
$\widetilde{\sigma}_{stk_Y}$
$ET_l \,\widetilde{\bowtie}_{jtk}\, ET_r$

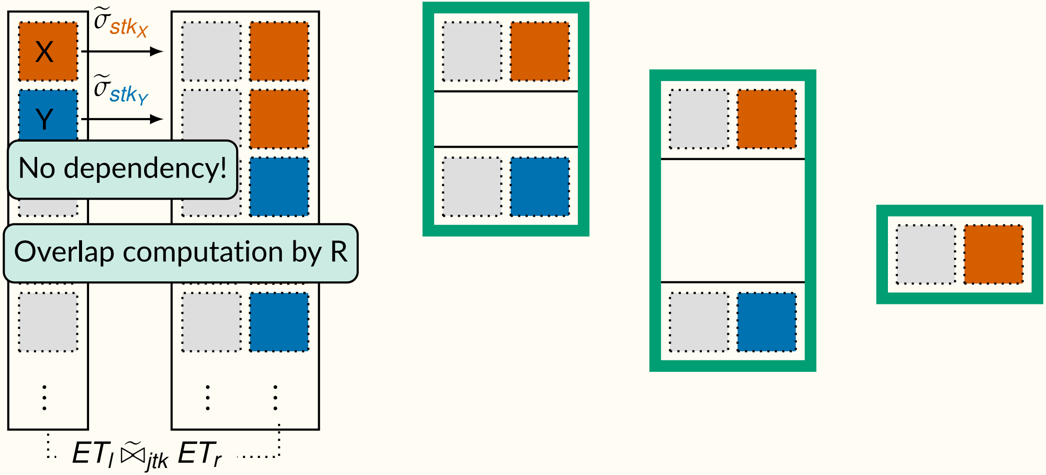
No dependency!

Overlap computation by R

Fixed-Point Operator

Run: $\bigcup_{R\in Z^+} \sigma_{S=F_{stk}(R)}$

SIMD (Single Instruction Multiple Data)

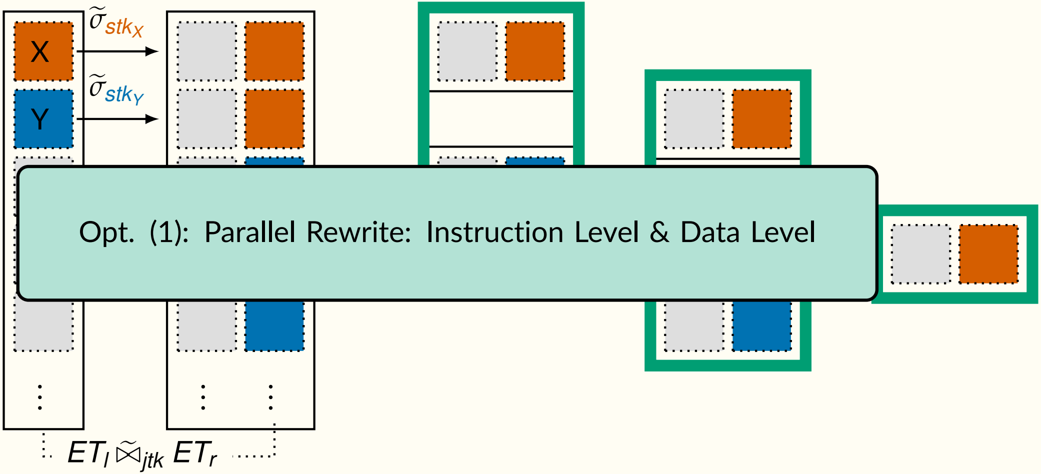

X
Y
$\widetilde{\sigma}_{stk_X}$
$\widetilde{\sigma}_{stk_Y}$
$ET_l \widetilde{\bowtie}_{jtk} ET_r$

No dependency!

Overlap computation by R

Fixed-Point Operator

Run: $\bigcup_{R\in Z^+} \sigma_{S=F_{stk}(R)}$

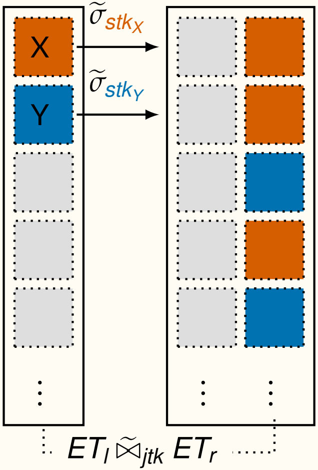

For $R = 1$
For $R = 2$
For $R = 3$
X
Y
$\widetilde{\sigma}_{stk_X}$
$\widetilde{\sigma}_{stk_Y}$
$ET_l \,\widetilde{\bowtie}_{jtk}\, ET_r$

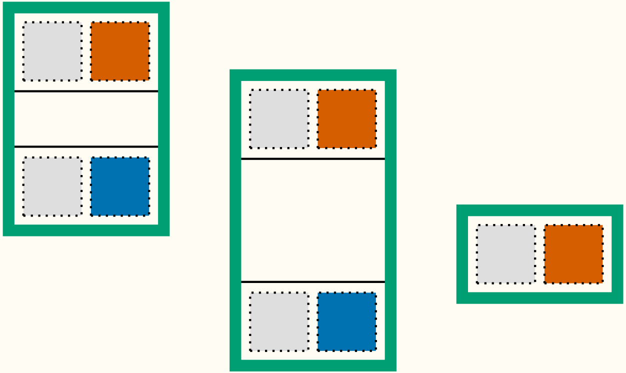
Opt. (1): Parallel Rewrite: Instruction Level & Data Level

Fixed-Point Operator

Run: $\bigcup_{R \in Z^+} \sigma_{S=F_{stk}(R)}$

Data Skew ⇒ Load Inbalance

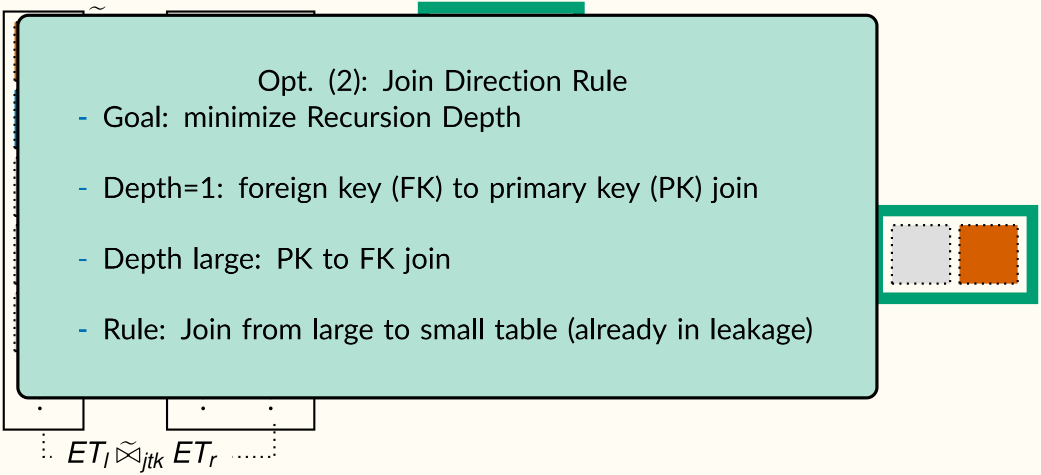

X
Y
$\widetilde{\sigma}_{stk_X}$
$\widetilde{\sigma}_{stk_Y}$
$ET_l \,\widetilde{\bowtie}_{jtk}\, ET_r$



Fixed-Point Operator

Run: $\bigcup_{R\in Z^+} \sigma_{S=F_{stk}(R)}$

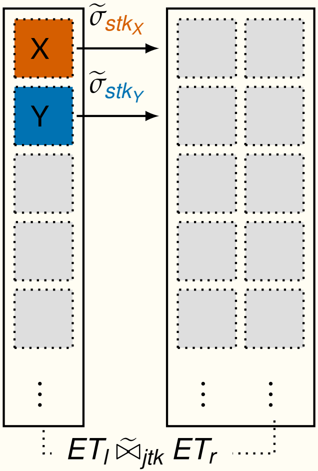
Data Skew ⇒ Load Inbalance



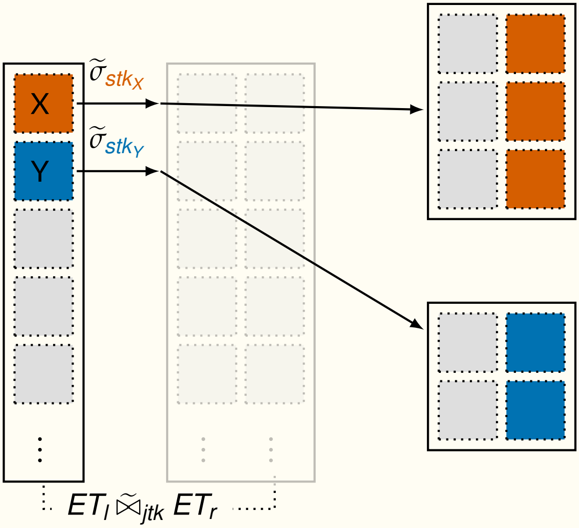
Opt. (2): Join Direction Rule

- Goal: minimize Recursion Depth
- Depth=1: foreign key (FK) to primary key (PK) join
- Depth large: PK to FK join
- Rule: Join from large to small table (already in leakage)

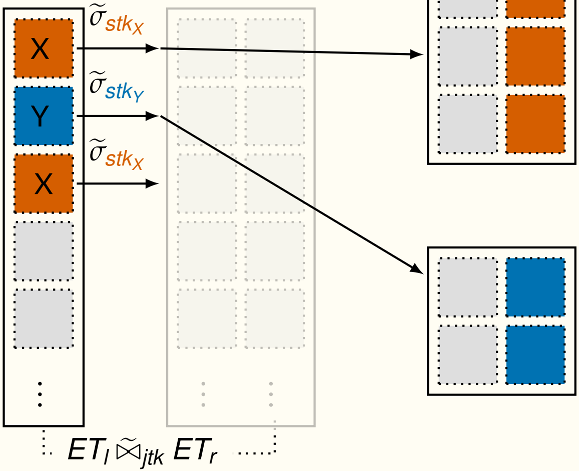
Fixed-Point Operator


X
Y
$\widetilde{\sigma}_{stk_X}$
$\widetilde{\sigma}_{stk_Y}$
$ET_l \,\widetilde{\bowtie}_{jtk}\, ET_r$

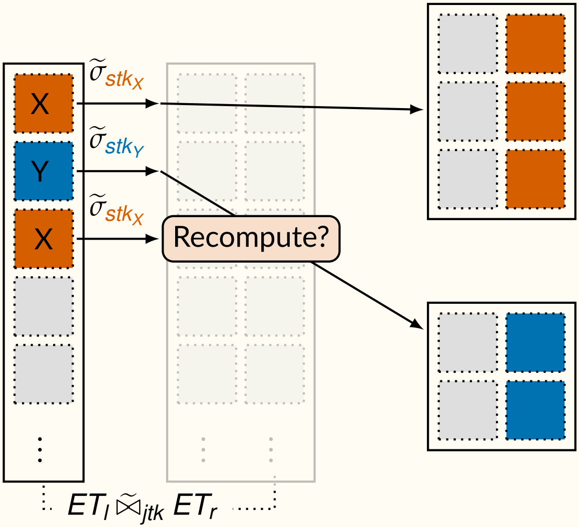
Fixed-Point Operator


X
Y
$\widetilde{\sigma}_{stk_X}$
$\widetilde{\sigma}_{stk_Y}$
$ET_l \widetilde{\bowtie}_{jtk} ET_r$

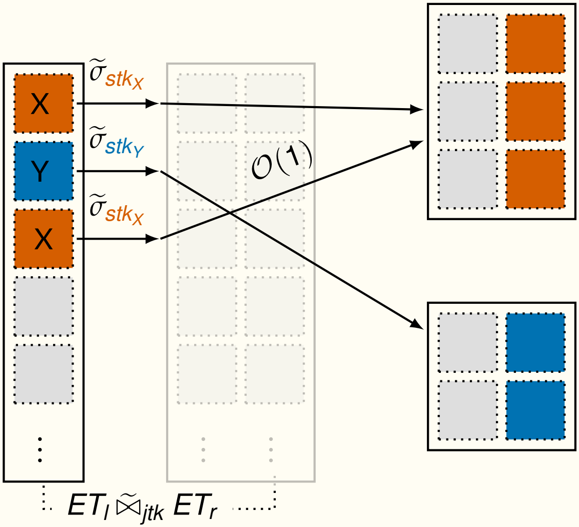
Fixed-Point Operator


X
Y
X
$\widetilde{\sigma}_{stk_X}$
$\widetilde{\sigma}_{stk_Y}$
$\widetilde{\sigma}_{stk_X}$
$ET_l \,\widetilde{\bowtie}_{jtk}\, ET_r$

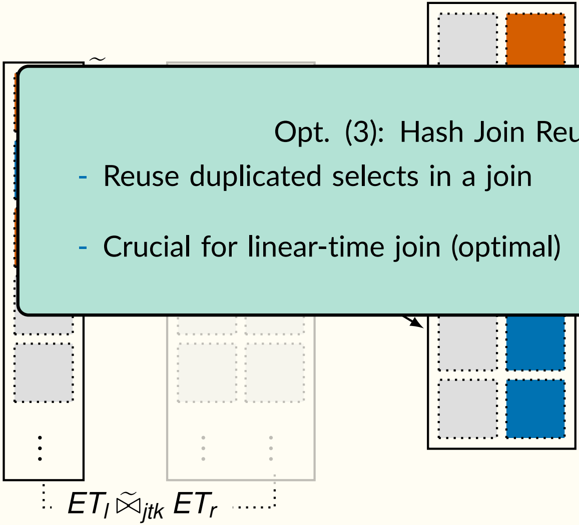
Fixed-Point Operator


X
Y
X
⋮
$\widetilde{\sigma}_{stk_X}$
$\widetilde{\sigma}_{stk_Y}$
$\widetilde{\sigma}_{stk_X}$
Recompute?
$ET_l \widetilde{\bowtie}_{jtk} ET_r$

Fixed-Point Operator


X
Y
X
$\widetilde{\sigma}_{stk_X}$
$\widetilde{\sigma}_{stk_Y}$
$\widetilde{\sigma}_{stk_X}$
$\mathcal{O}(1)$
$ET_l \,\widetilde{\bowtie}_{jtk}\, ET_r$

Fixed-Point Operator



Opt. (3): Hash Join Reuse

- Reuse duplicated selects in a join

- Crucial for linear-time join (optimal)