Side Channel Attacks on Data Processing Applications

Outline

- Overview of cache side-channel attacks
- Database Reconstruction from Noisy Volumes: A Cache Side-Channel Attack on SQLite.
  A. Shahverdi, M. Shirinov, D. Dachman-Soled.
  USENIX 2021
- How to Own the NAS in Your Spare Time.
  S. Hong, M. Davinroy, Y. Kaya, D. Dachman-Soled, T. Dumitras.
  ICLR 2020
  - Security analysis of deep neural networks operating in the presence of cache side-channel attacks.

  Sanghyun Hong, Michael Davinroy, Yiğitcan Kaya, Stuart Nevans Locke, Ian Rackow, Kevin Kulda, Dana Dachman-Soled, Tudor Dumitras, arXiv 2018.

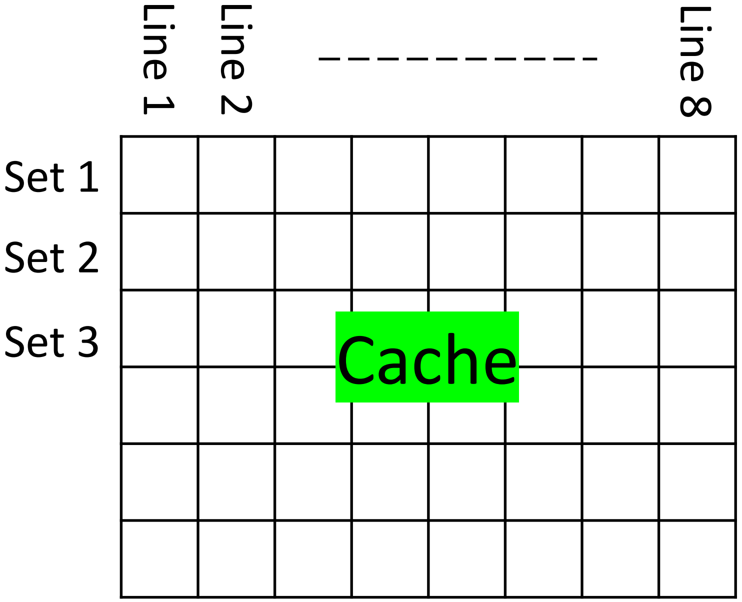
Outline

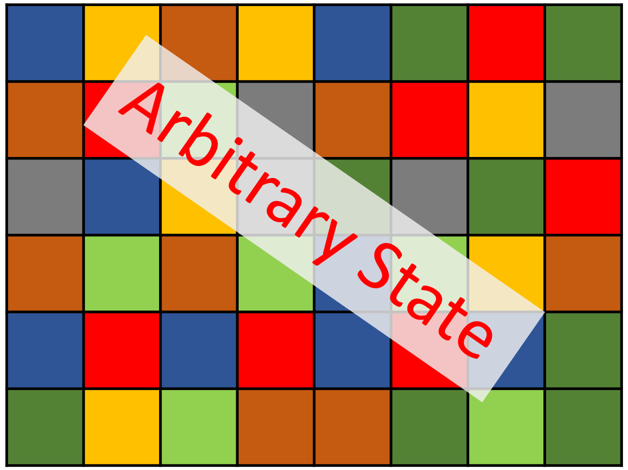
- Overview of cache side-channel attacks
- Database Reconstruction from Noisy Volumes: A Cache Side-Channel Attack on SQLite.
  A. Shahverdi, M. Shirinov, D. Dachman-Soled.
  USENIX 2021
- How to Own the NAS in Your Spare Time.
  S. Hong, M. Davinroy, Y. Kaya, D. Dachman-Soled, T. Dumitras.
  ICLR 2020
  - Security analysis of deep neural networks operating in the presence of cache side-channel attacks.

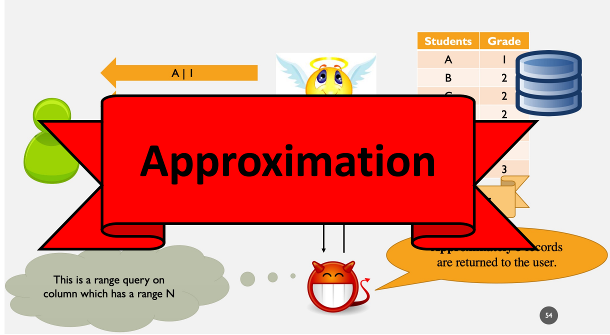
  Sanghyun Hong, Michael Davinroy, Yiğitcan Kaya, Stuart Nevans Locke, Ian Rackow, Kevin Kulda, Dana Dachman-Soled, Tudor Dumitras, arXiv 2018.

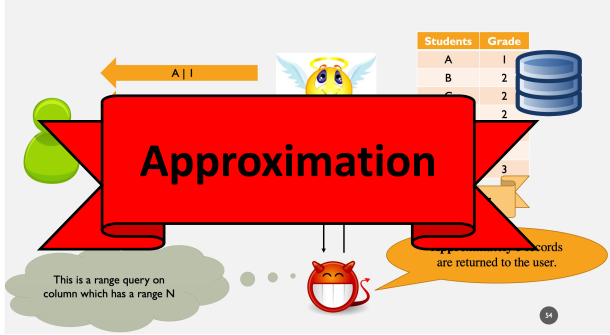
Flush and Reload

1. Flush memory line
2. Wait a bit
3. Measure time to Reload line
4. Repeat



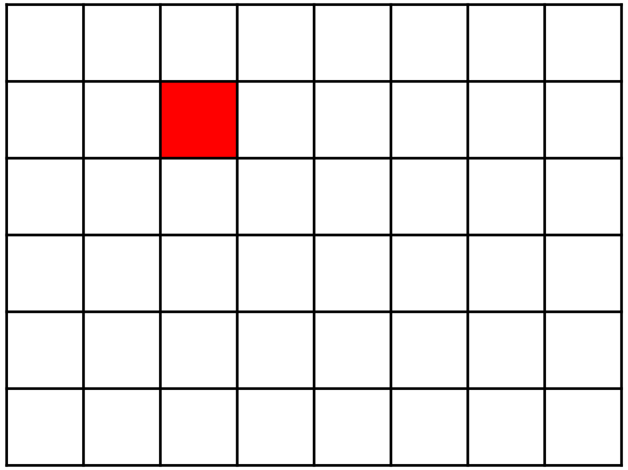


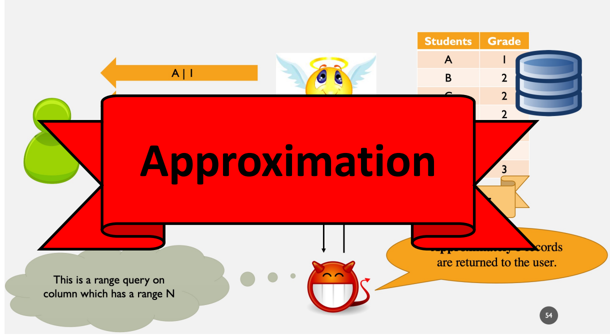

Approximation
A | I
Students
Grade
A
I
B
2
2
2
3
This is a range query on column which has a range N
...cords are returned to the user.
54


Students
Grade
A
1
B
2
2
2
3
A | 1
Approximation
This is a range query on column which has a range N
cords
are returned to the user.
54

Flush and Reload

1. Flush a memory line
2. Wait a bit
3. Measure time to Reload line
4. Repeat





Students	Grade
A	I
B	2
[illegible]	2
[illegible]	2
[illegible]	3

A | I

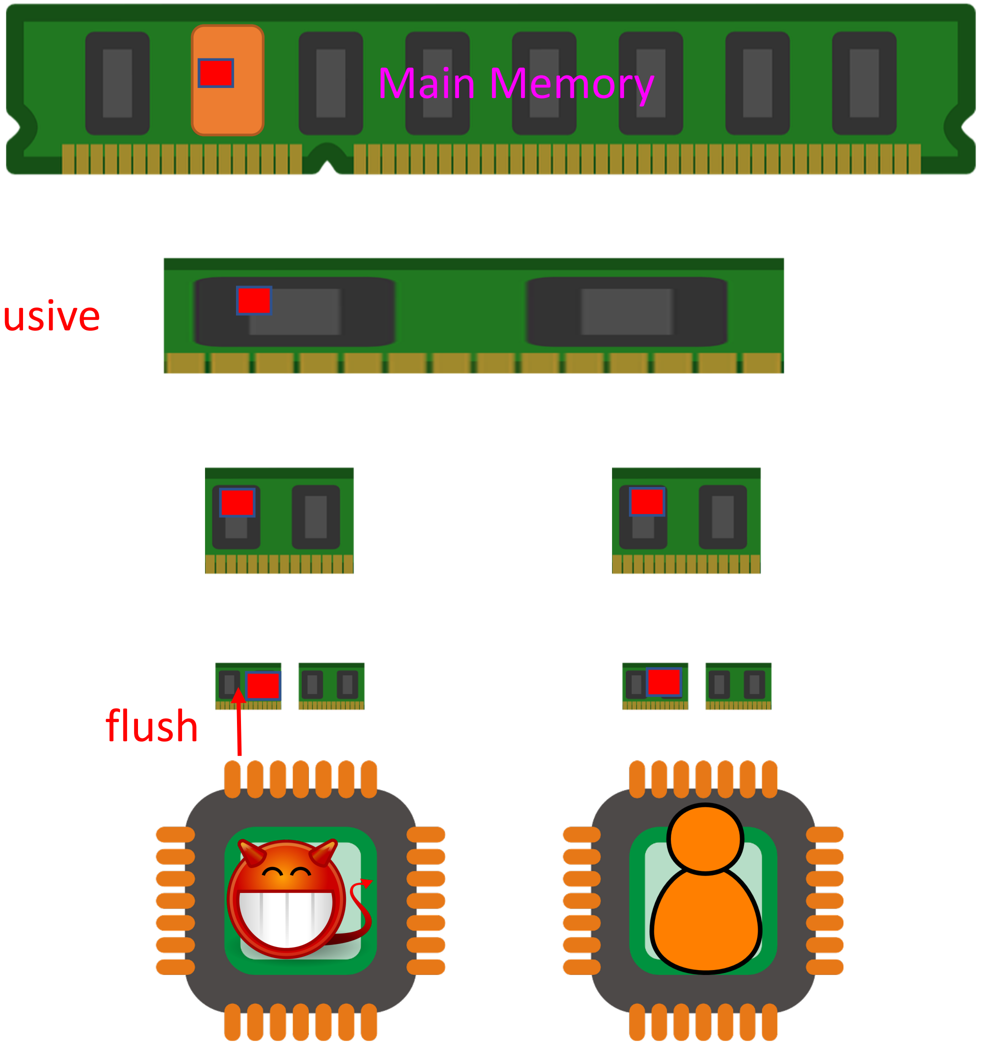
Approximation

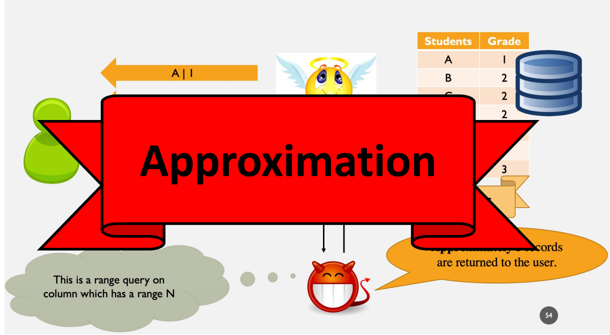
This is a range query on column which has a range N

[illegible] cords are returned to the user.

Flush a Line From Cache

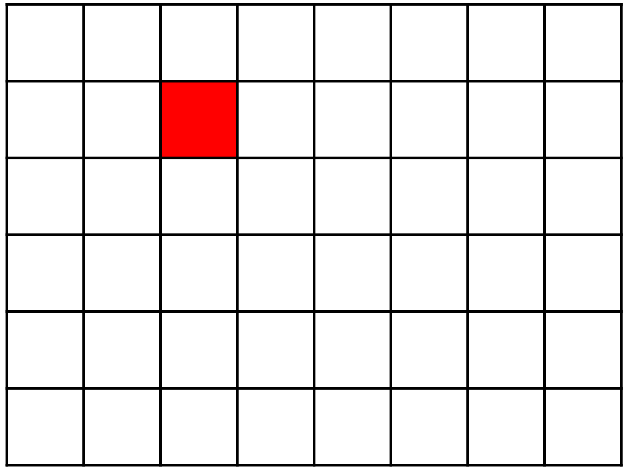
Last Level Cache (LLC) is inclusive



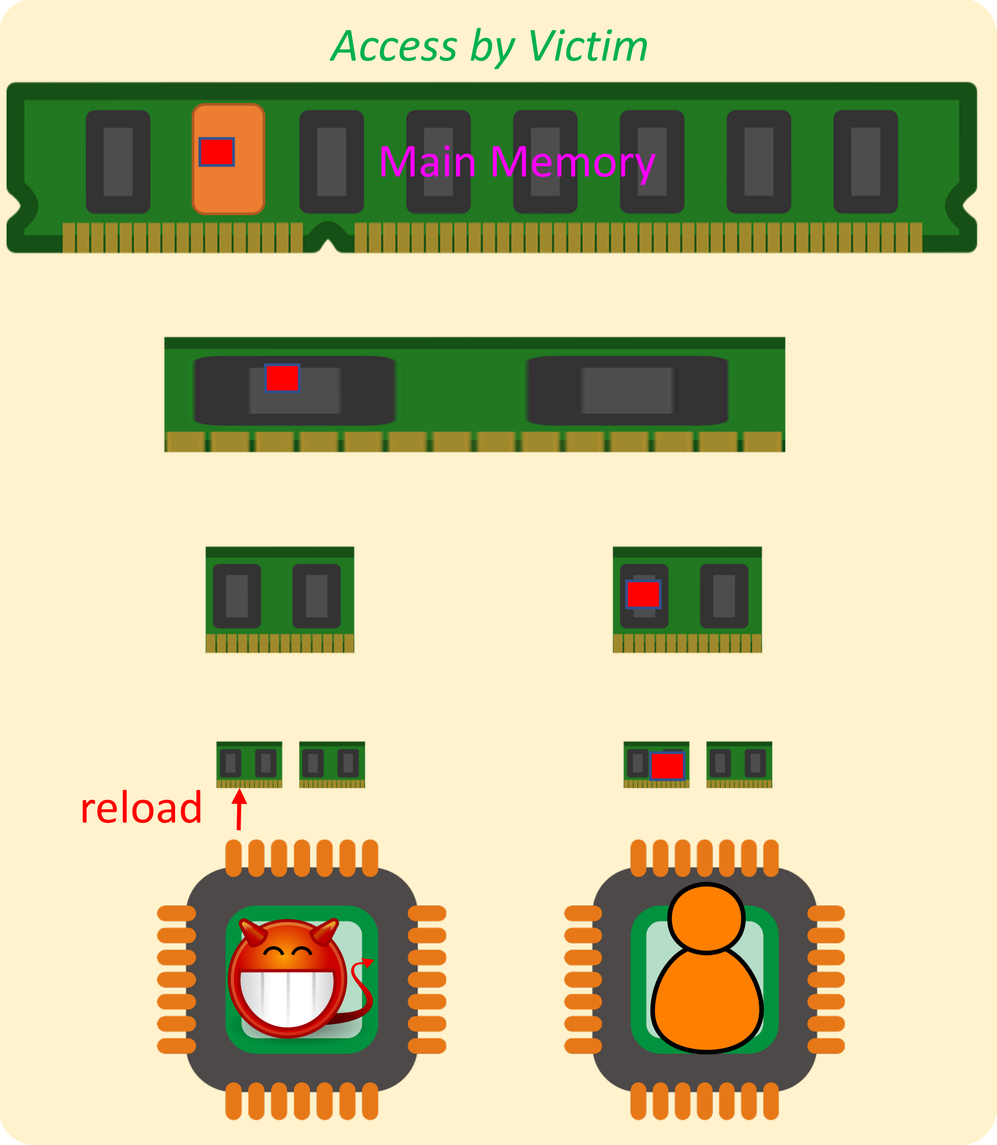

A | I
Students
Grade
A
I
B
2
2
2
3
Approximation
This is a range query on column which has a range N
are returned to the user.
54

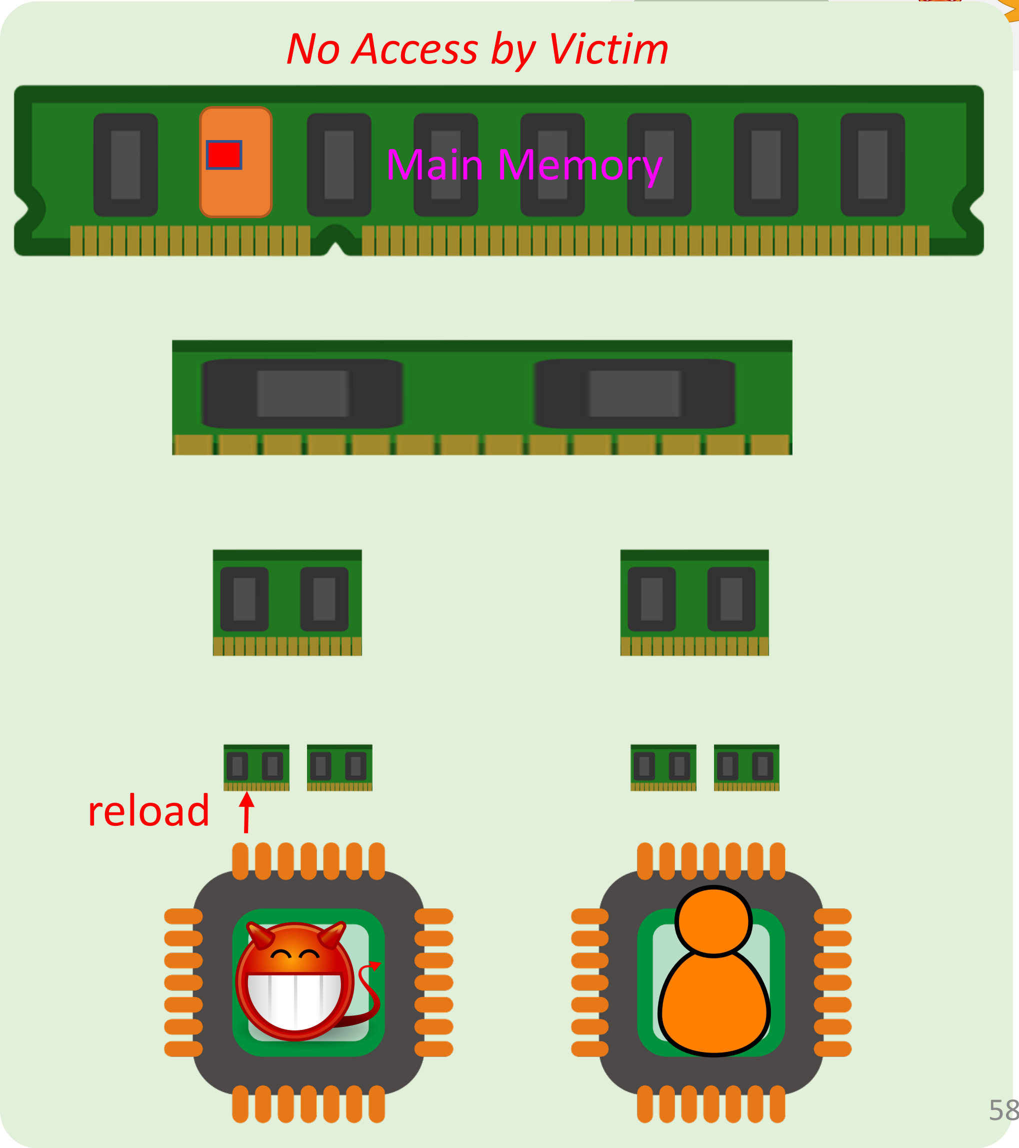
Flush and Reload

1. Flush a memory line
2. Wait a bit
3. Measure time to Reload line
4. Repeat

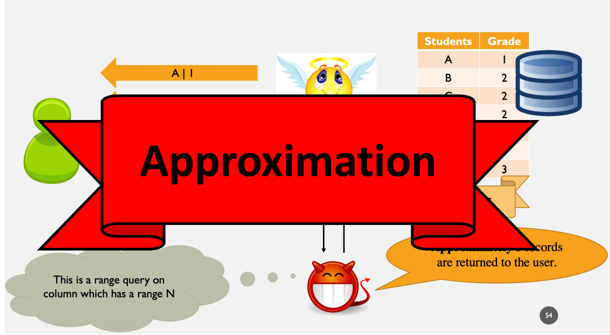


Reload a Line From Cache





Flush and Reload



Students	Grade
A	1
B	2
C	2
	2
	3

A | I

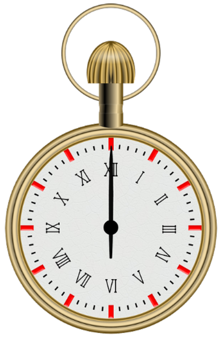
Approximation

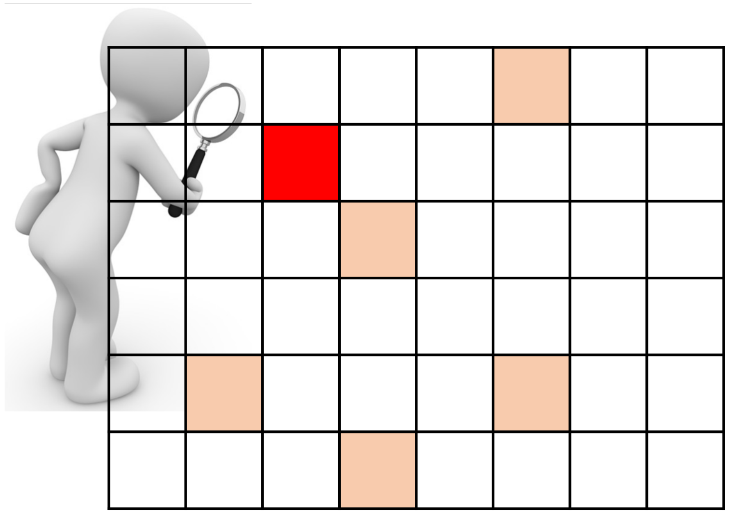
This is a range query on column which has a range N

…cords are returned to the user.

54

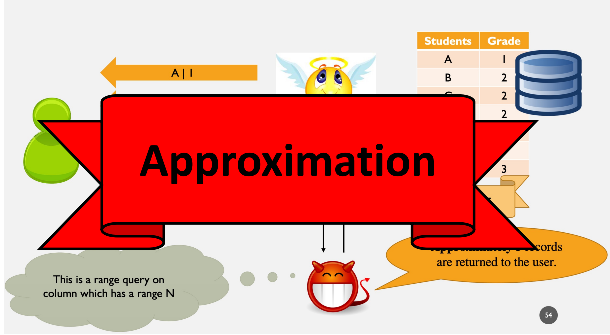
1. Flush memory line
2. Wait a bit
3. Measure time to Reload line
4. Repeat





Slow means no access by victim

Fast means that victim accessed

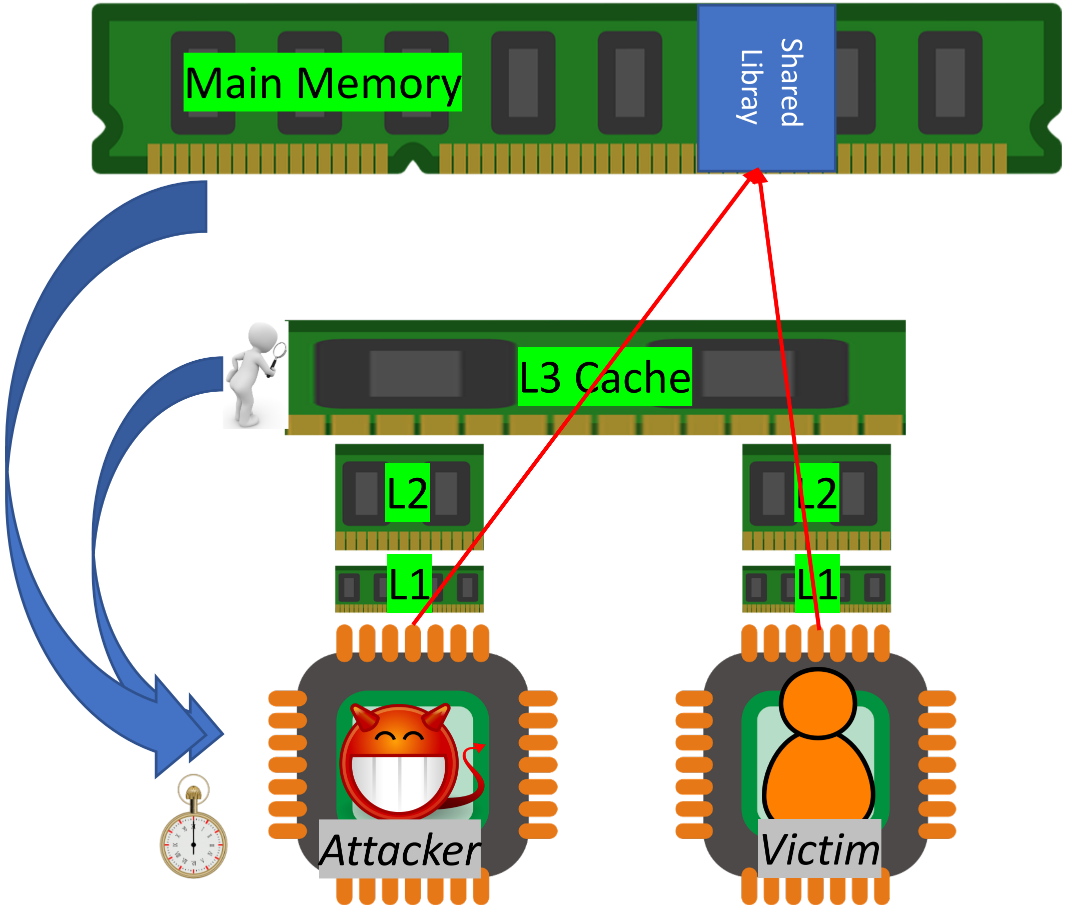


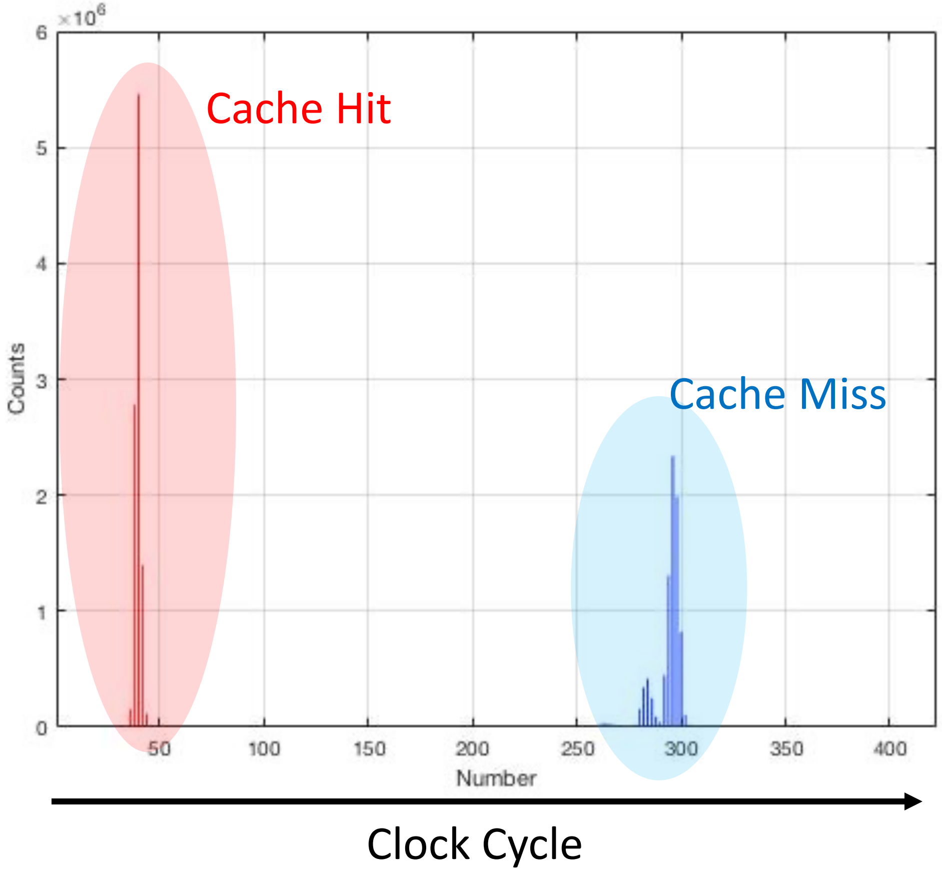
Approximation

This is a range query on column which has a range N

are returned to the user.

Cache Attack Summary

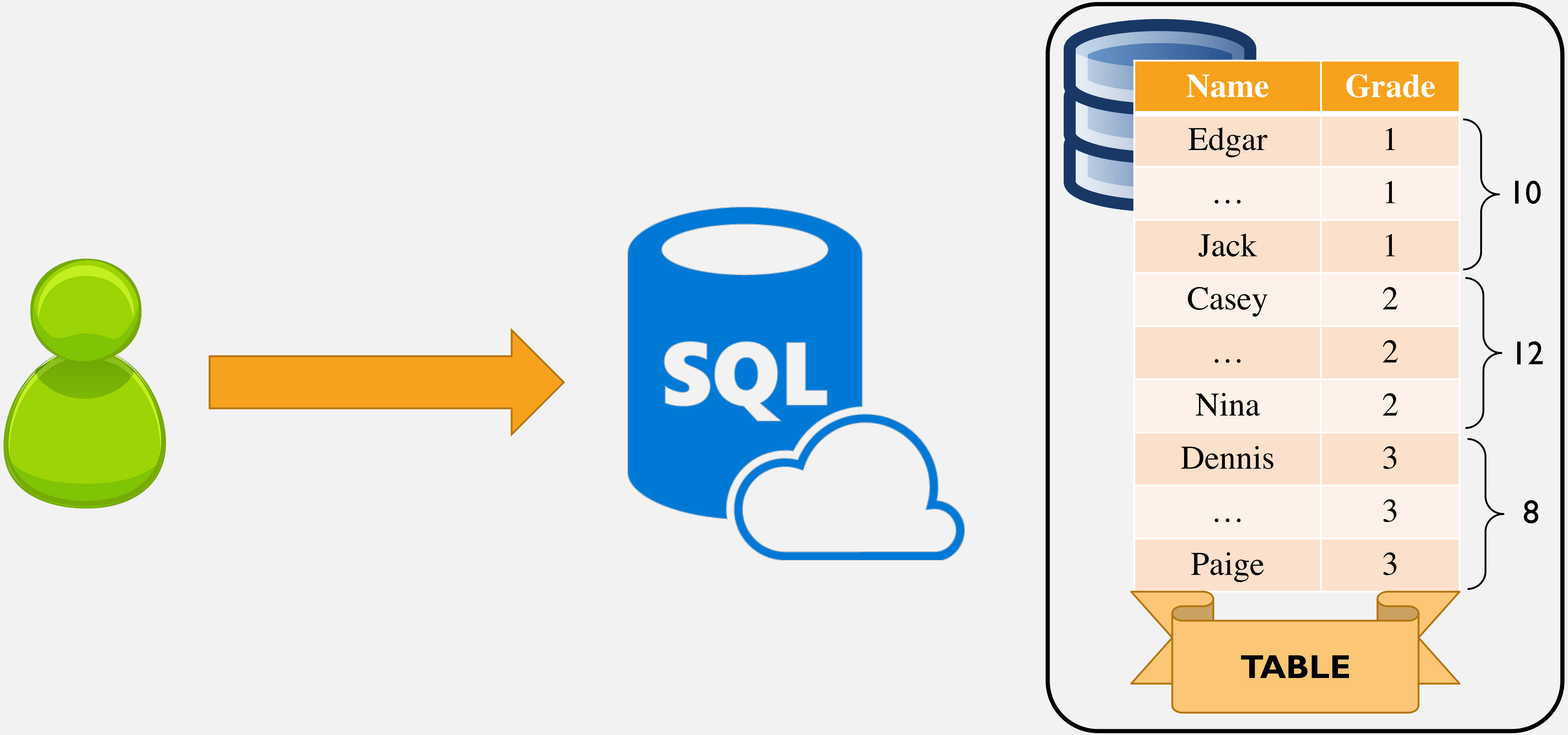




Outline

- Overview of cache side-channel attacks
- Database Reconstruction from Noisy Volumes: A Cache Side-Channel Attack on SQLite.
  A. Shahverdi, M. Shirinov, D. Dachman-Soled.
  USENIX 2021
- How to Own the NAS in Your Spare Time.
  S. Hong, M. Davinroy, Y. Kaya, D. Dachman-Soled, T. Dumitras.
  ICLR 2020
  - Security analysis of deep neural networks operating in the presence of cache side-channel attacks.

    Sanghyun Hong, Michael Davinroy, Yiğitcan Kaya, Stuart Nevans Locke, Ian Rackow, Kevin Kulda, Dana Dachman-Soled, Tudor Dumitras, arXiv 2018.

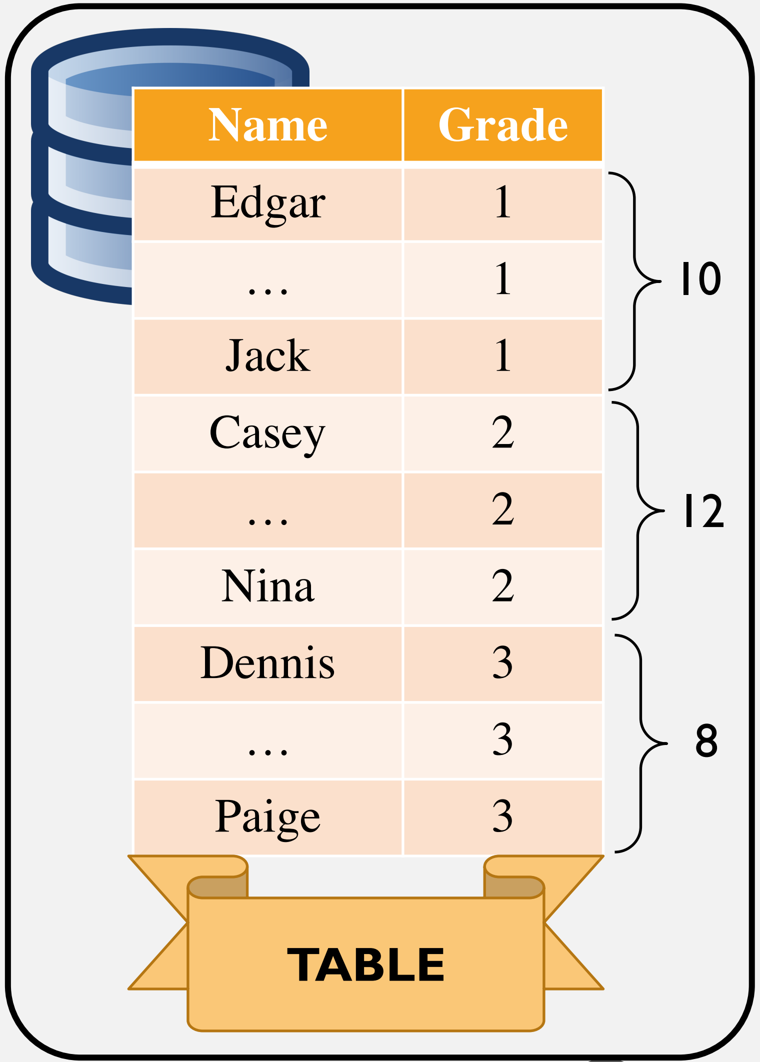


Name	Grade
Edgar	1
…	1
Jack	1
Casey	2
…	2
Nina	2
Dennis	3
…	3
Paige	3

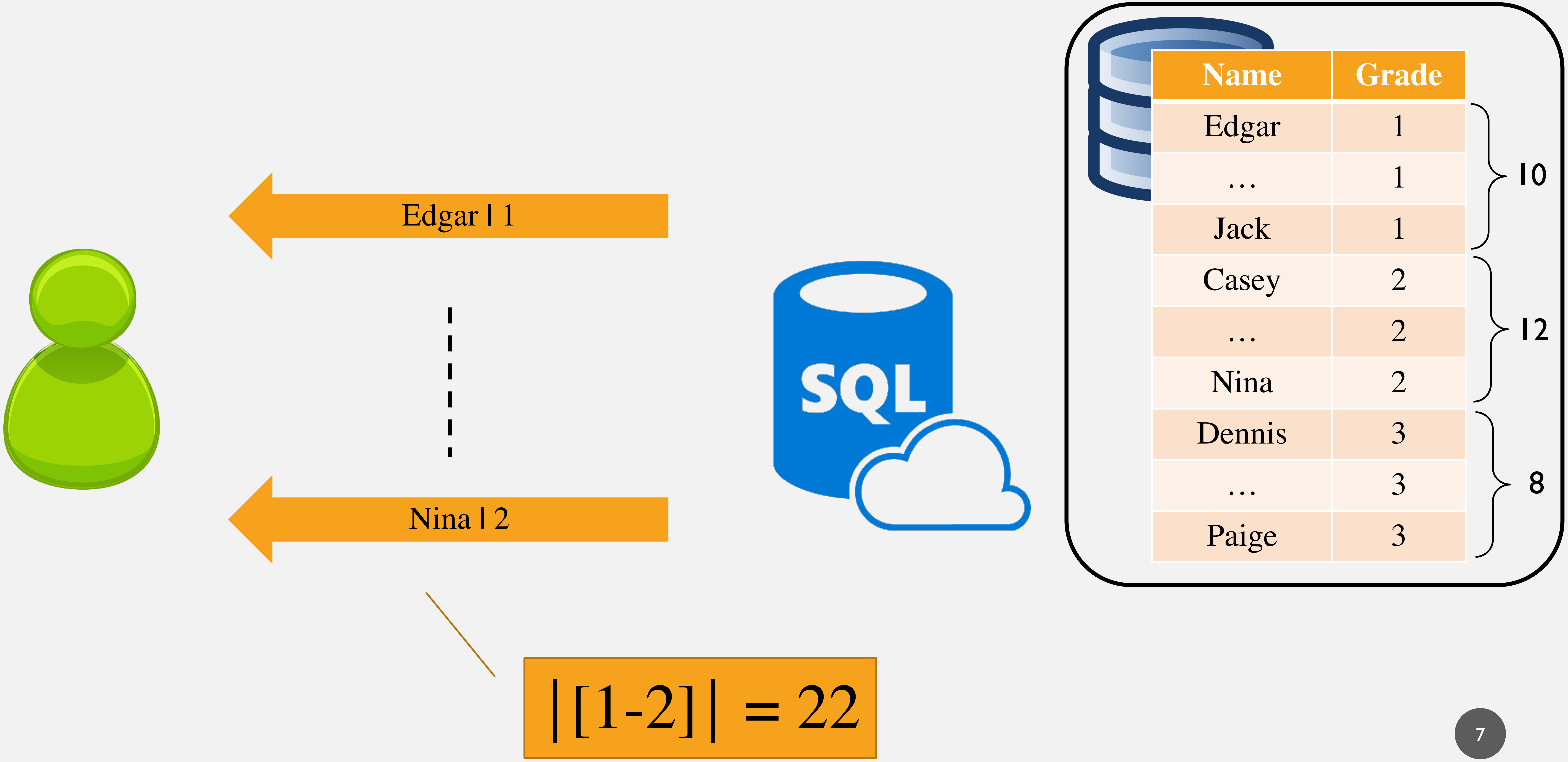
Range Query

SELECT *
FROM *TABLE*
*WHERE **GRADE***
*BETWEEN **1 AND 2***

[1-2]



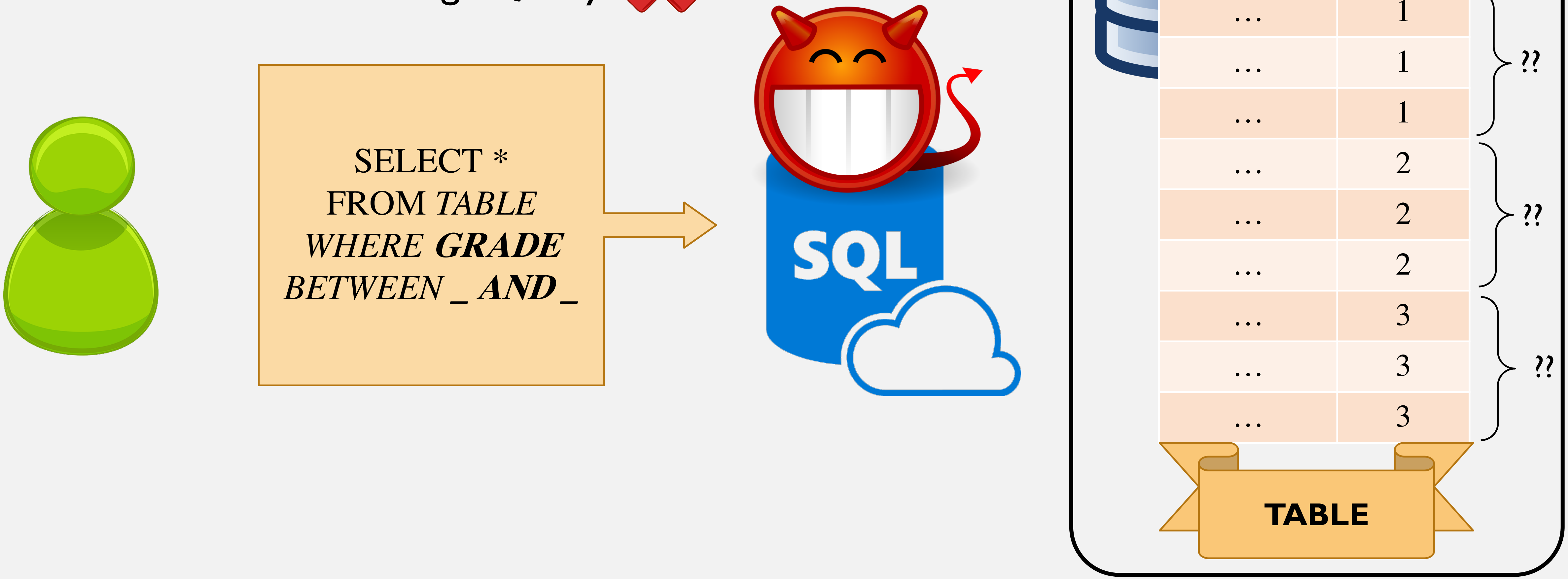
Name	Grade
Edgar	1
…	1
Jack	1
Casey	2
…	2
Nina	2
Dennis	3
…	3
Paige	3



Name	Grade	
Edgar	1	10
…	1	
Jack	1	
Casey	2	12
…	2	
Nina	2	
Dennis	3	8
…	3	
Paige	3	

$|[1\text{-}2]| = 22$

Query ✓
Range Query ✓
Column Name ✓
Range of the Column ✓
Content of Range Query ✗



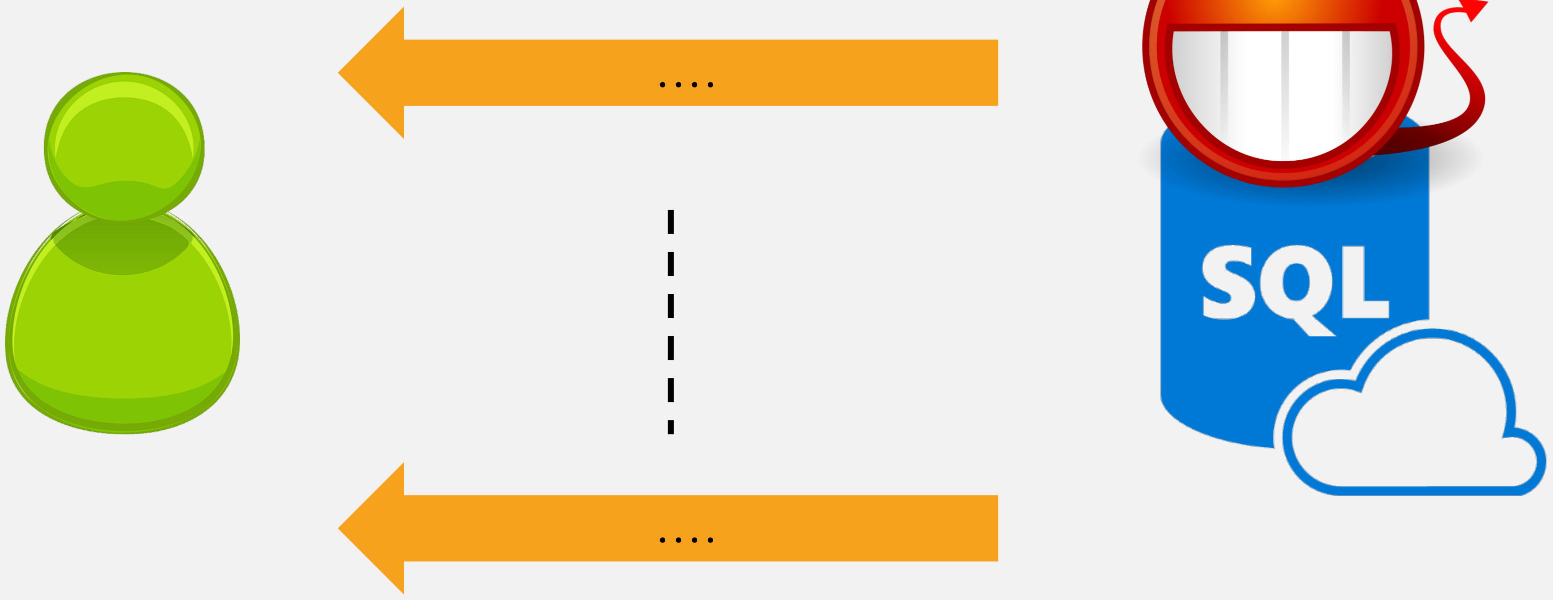
SELECT *
FROM *TABLE*
WHERE ***GRADE***
*BETWEEN _ **AND** _*

…	Grade
…	1
…	1
…	1
…	2
…	2
…	2
…	3
…	3
…	3

22 Records from the database is returned to the user

Number of Entries ✓

Content of Entries ✗

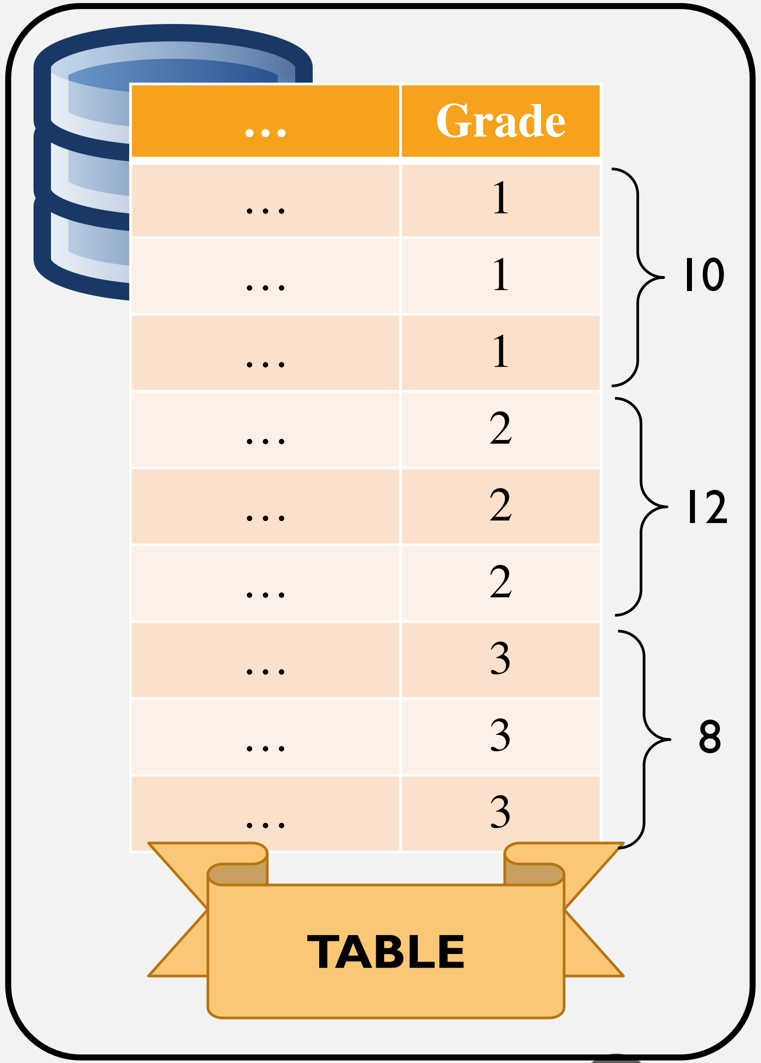


...	Grade	
...	1	
...	1	??
...	1	
...	2	
...	2	??
...	2	
...	3	
...	3	??
...	3	

TABLE

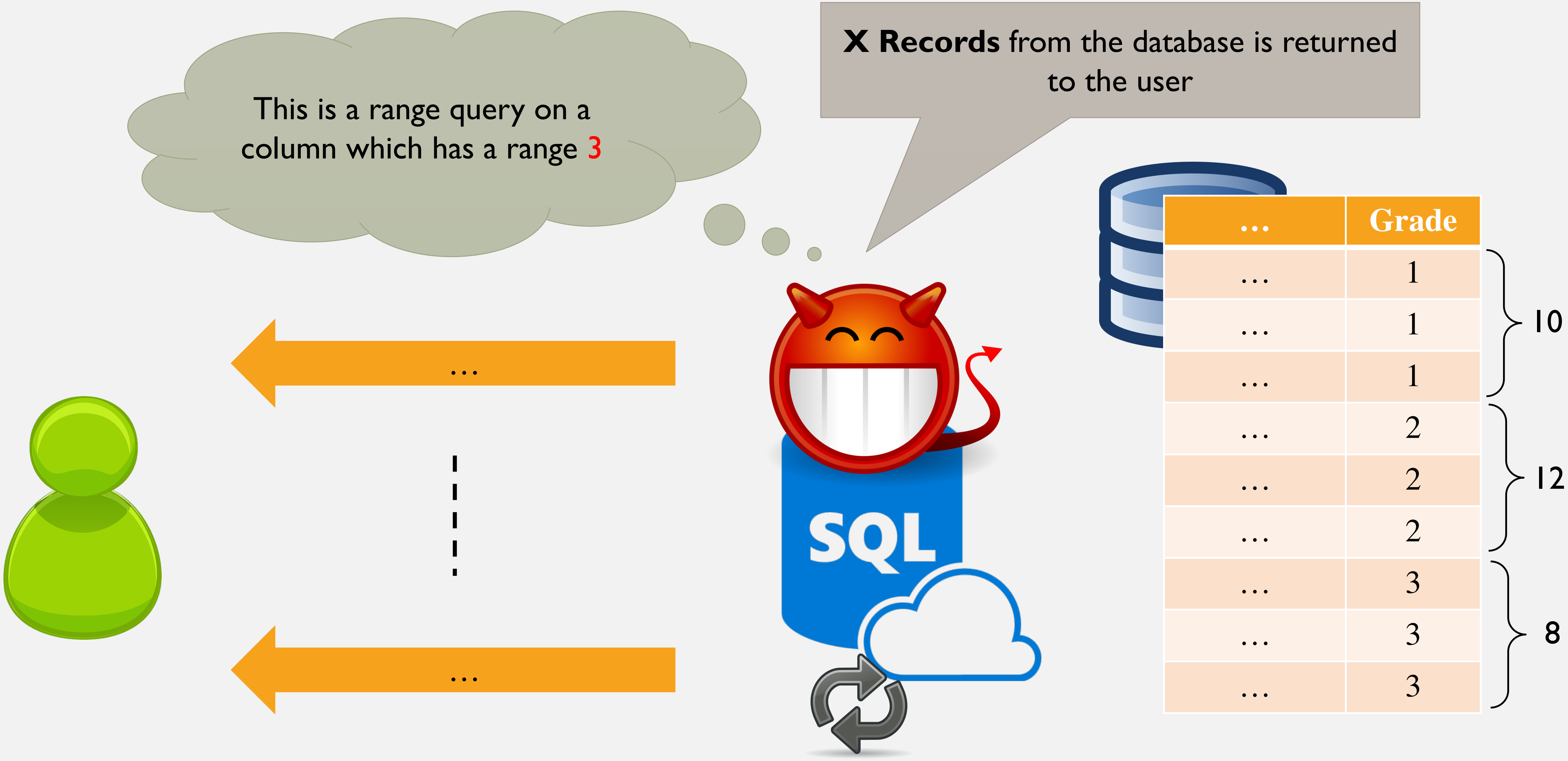
Grade	
1	10
1	
1	
2	12
2	
2	
3	8
3	
3	

Reconstructed Database



…	Grade	
…	1	10
…	1	
…	1	
…	2	12
…	2	
…	2	
…	3	8
…	3	
…	3	

TABLE



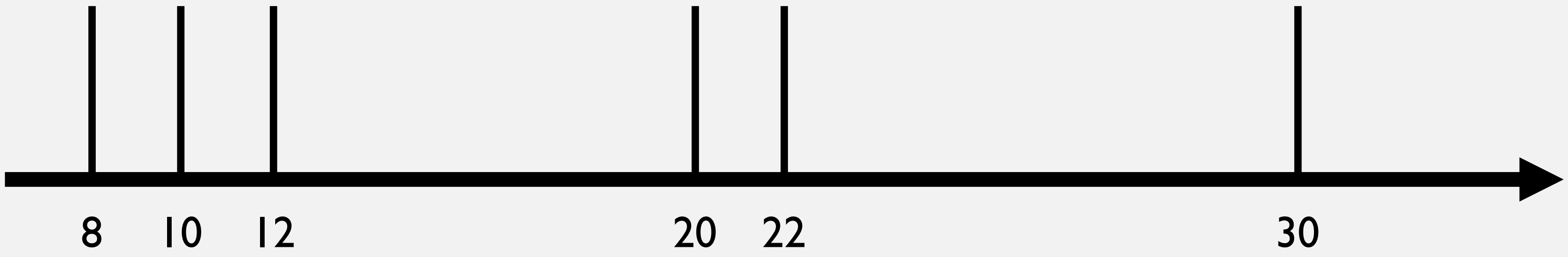
This is a range query on a column which has a range 3

X Records from the database is returned to the user

…	Grade
…	1
…	1
…	1
…	2
…	2
…	2
…	3
…	3
…	3

Name	Grade	
Edgar	1	10
…	1	
Jack	1	
Casey	2	12
…	2	
Nina	2	
Dennis	3	8
…	3	
Paige	3	

	[1-1]	[2-2]	[3-3]	[1-2]	[2-3]	[1-3]
Volume						

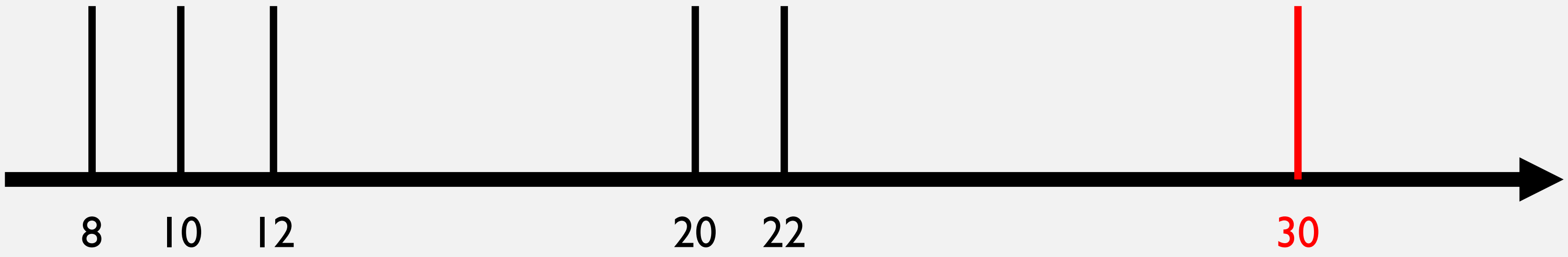


Range = 3

1. Maximum Value is 30 so it has to be for range [1-3]
2. |[1-3]| = 30

	[1-1]	[2-2]	[3-3]	[1-2]	[2-3]	[1-3]
Volume						

Name	Grade	
Edgar	1	10
…	1	
Jack	1	
Casey	2	12
…	2	
Nina	2	
Dennis	3	8
…	3	
Paige	3	

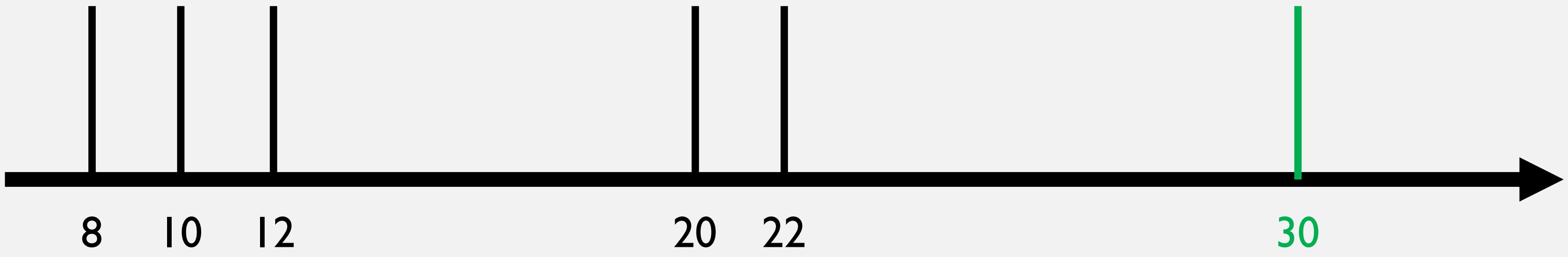


Range = 3

1. Maximum Value is 30 so it has to be for range [1-3]
2. |[1-3]| = 30

	[1-1]	[2-2]	[3-3]	[1-2]	[2-3]	~~[1-3]~~
Volume						30

Name	Grade	
Edgar	1	10
…	1	
Jack	1	
Casey	2	12
…	2	
Nina	2	
Dennis	3	8
…	3	
Paige	3	

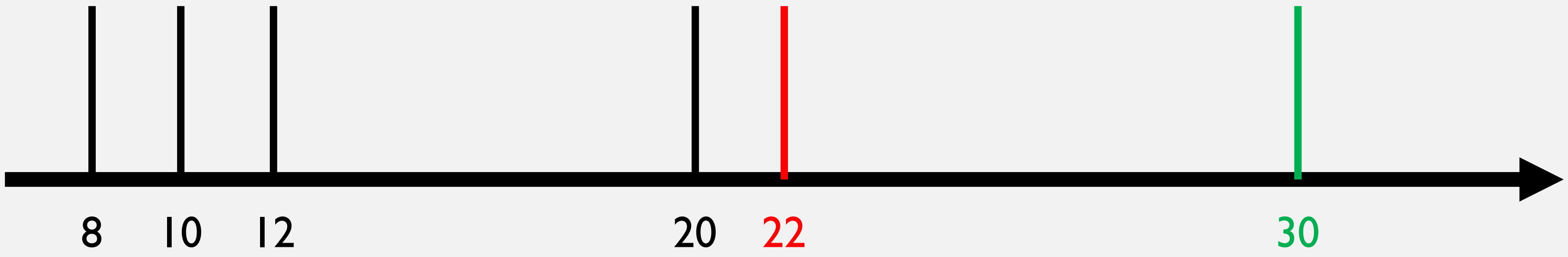


Range = 3

1. Maximum Value is 30 so it has to be for range [1-3]
2. |[1-3]| = 30
3. The next biggest range is [1-2] or [2-3]
4. Let’s say |[1-2]| = 22

	[1-1]	[2-2]	[3-3]	[1-2]	[2-3]	~~[1-3]~~
Volume						30

Name	Grade	
Edgar	1	10
…	1	
Jack	1	
Casey	2	12
…	2	
Nina	2	
Dennis	3	8
…	3	
Paige	3	

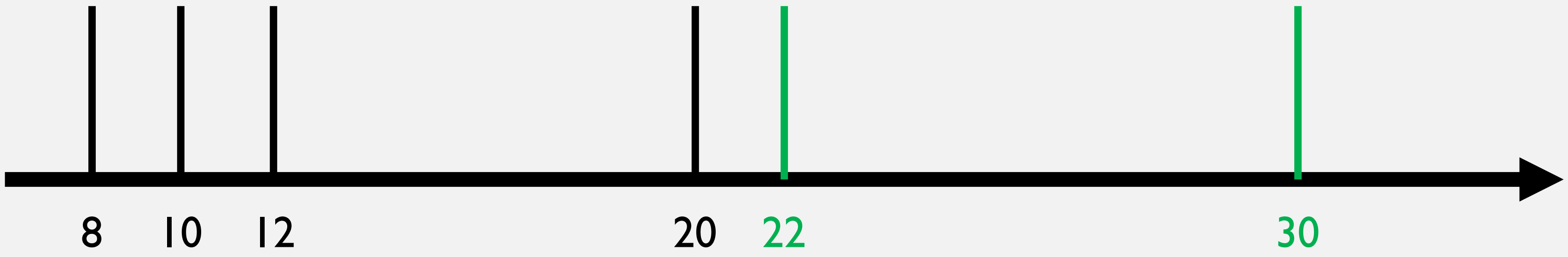


Range = 3

1. Maximum Value is 30 so it has to be for range [1-3]
2. |[1-3]| = 30
3. The next biggest range is [1-2] or [2-3]
4. Let’s say |[1-2]| = 22

Name	Grade	
Edgar	1	10
…	1	
Jack	1	
Casey	2	12
…	2	
Nina	2	
Dennis	3	8
…	3	
Paige	3	

	[1-1]	[2-2]	[3-3]	~~[1-2]~~	[2-3]	~~[1-3]~~
Volume				22		30

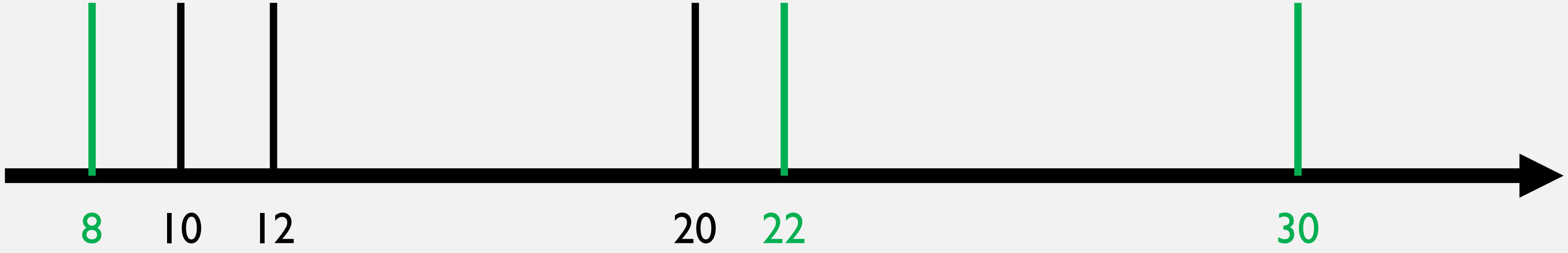


Range = 3

1. Maximum Value is 30 so it has to be for range [1-3]
2. |[1-3]| = 30
3. The next biggest range is [1-2] or [2-3]
4. Let’s say |[1-2]| = 22
5. Then Because |[1-3]| = 30 it has to be |[3-3]| = 8

	[1-1]	[2-2]	~~[3-3]~~	~~[1-2]~~	[2-3]	~~[1-3]~~
Volume			8	22		30

Name	Grade	
Edgar	1	10
…	1	
Jack	1	
Casey	2	12
…	2	
Nina	2	
Dennis	3	8
…	3	
Paige	3	

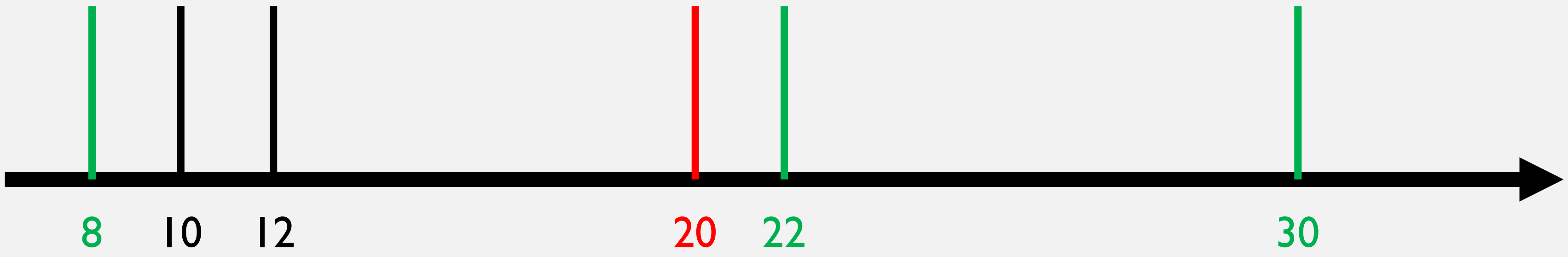


Range = 3

1. Maximum Value is 30 so it has to be for range [1-3]
2. |[1-3]| = 30
3. The next biggest range is [1-2] or [2-3].
4. Let’s say |[1-2]| = 22
5. Then Because |[1-3]| = 30 it has to be |[3-3]| = 8
6. The next biggest range is [2-3] and its volume is 20.

	[1-1]	[2-2]	~~[3-3]~~	~~[1-2]~~	[2-3]	~~[1-3]~~
Volume			8	22		30

Name	Grade	
Edgar	1	10
…	1	
Jack	1	
Casey	2	12
…	2	
Nina	2	
Dennis	3	8
…	3	
Paige	3	

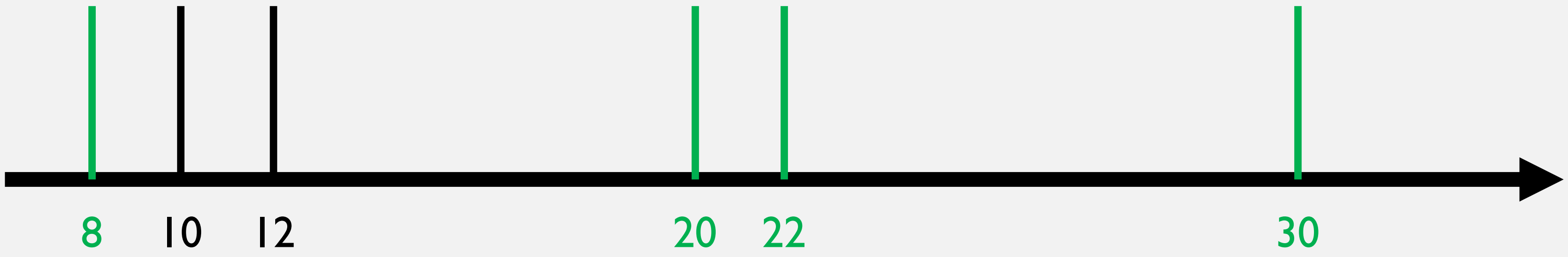


Range = 3

1. Maximum Value is 30 so it has to be for range [1-3]
2. |[1-3]| = 30
3. The next biggest range is [1-2] or [2-3].
4. Let’s say |[1-2]| = 22
5. Then Because |[1-3]| = 30 it has to be |[3-3]| = 8
6. The next biggest range is [2-3] and its volume is 20.

	[1-1]	[2-2]	~~[3-3]~~	~~[1-2]~~	~~[2-3]~~	~~[1-3]~~
Volume			8	22	20	30

Name	Grade	
Edgar	1	10
…	1	
Jack	1	
Casey	2	12
…	2	
Nina	2	
Dennis	3	8
…	3	
Paige	3	

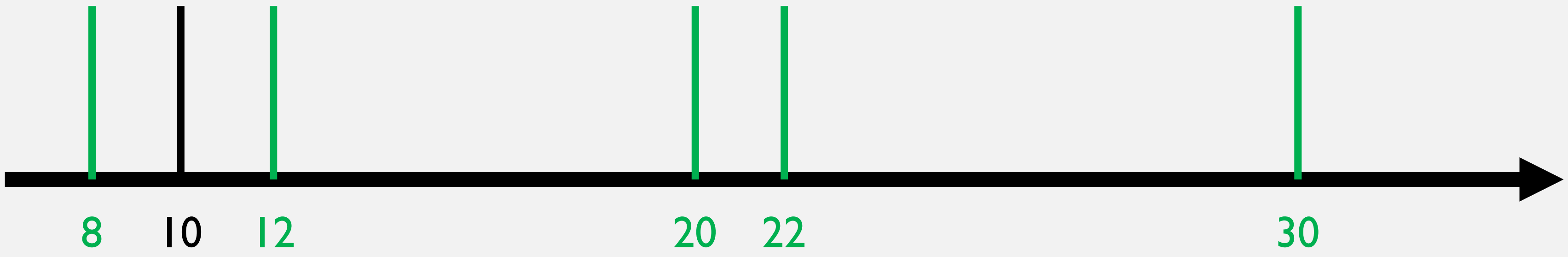


Range = 3

1. Maximum Value is 30 so it has to be for range [1-3]
2. |[1-3]| = 30
3. The next biggest range is [1-2] or [2-3].
4. Let's say |[1-2]| = 22
5. Then Because |[1-3]| = 30 it has to be |[3-3]| = 8
6. The next biggest range is [2-3] and its volume is 20.
7. We know |[3-3]| = 8 so it has to be |[2-2]| = 12

	[1-1]	~~[2-2]~~	~~[3-3]~~	~~[1-2]~~	~~[2-3]~~	~~[1-3]~~
Volume		12	8	22	20	30

Name	Grade	
Edgar	1	10
…	1	
Jack	1	
Casey	2	12
…	2	
Nina	2	
Dennis	3	8
…	3	
Paige	3	

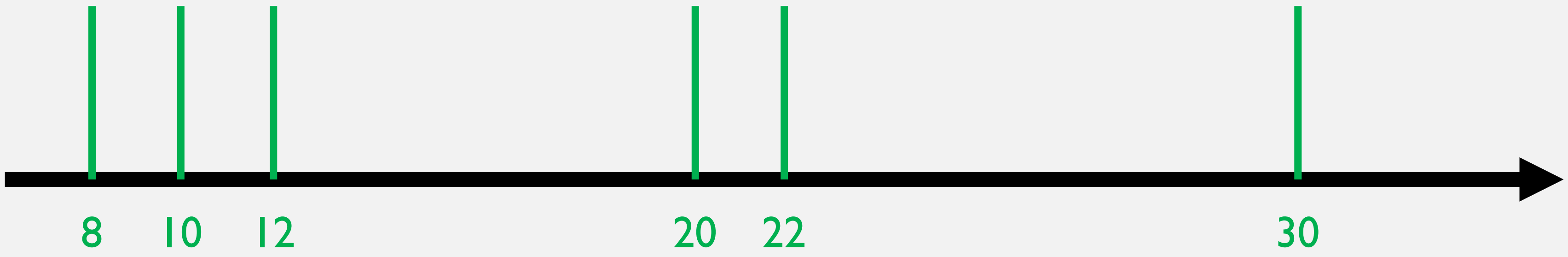


Range = 3

1. Maximum Value is 30 so it has to be for range [1-3]
2. |[1-3]| = 30
3. The next biggest range is [1-2] or [2-3].
4. Let’s say |[1-2]| = 22
5. Then Because |[1-3]| = 30 it has to be |[3-3]| = 8
6. The next biggest range is [2-3] and its volume is 20.
7. We know |[3-3]| = 8 so it has to be |[2-2]| = 12
8. There is only one range left so |[1-1]| = 10

	[1-1]	[2-2]	[3-3]	[1-2]	[2-3]	[1-3]
Volume	10	12	8	22	20	30

Name	Grade	
Edgar	1	10
…	1	
Jack	1	
Casey	2	12
…	2	
Nina	2	
Dennis	3	8
…	3	
Paige	3	



Range = 3

1. Maximum Value is 30 so it has to be for range [1-3]
2. |[1-3]| = 30
3. The next biggest range is [1-2] or [2-3].
4. Let’s say |[1-2]| = 22
5. Then Because |[1-3]| = 30 it has to be |[3-3]| = 8
6. The next biggest range is [2-3] and its volume is 20.
7. We know |[3-3]| = 8 so it has to be |[2-2]| = 12
8. There is only one range left so |[1-1]| = 10

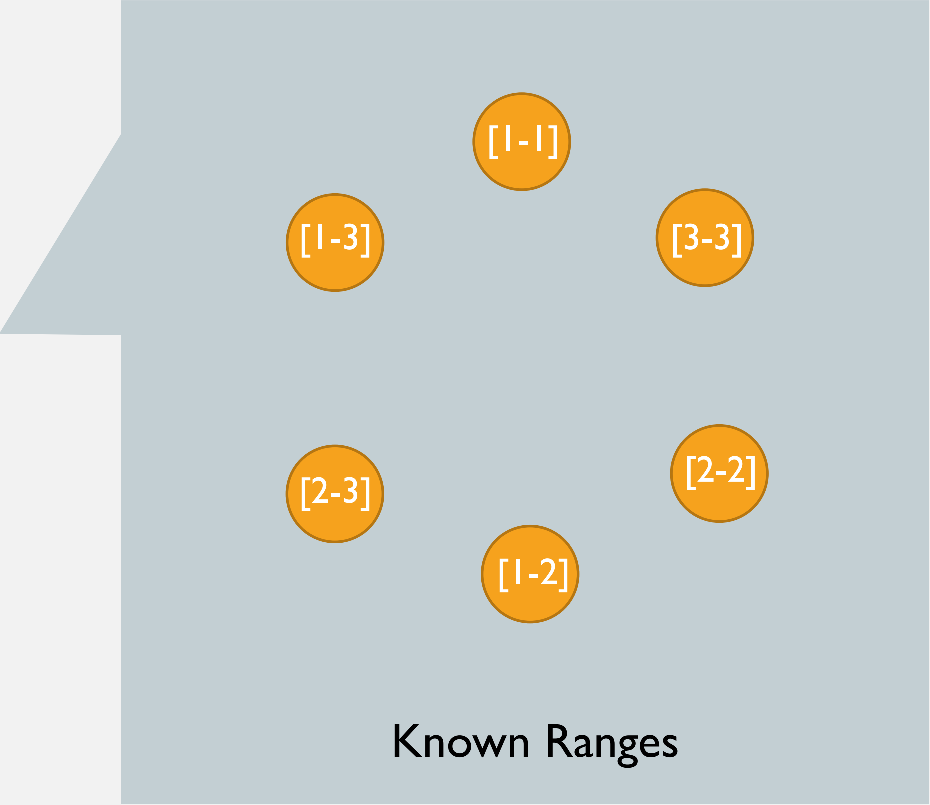
Grade	
1	10
…	
1	
2	12
…	
2	
3	8
…	
3	

Name	Grade	
Edgar	1	10
…	1	
Jack	1	
Casey	2	12
…	2	
Nina	2	
Dennis	3	8
…	3	
Paige	3	

<table>
<tr><th>|[1-1]|</th><th>|[2-2]|</th><th>|[3-3]|</th><th>|[1-2]|</th><th>|[2-3]|</th><th>|[1-3]|</th></tr>
<tr><td></td><td></td><td></td><td>|[1-1]| + |[2-2]|</td><td>|[2-2]| + |[3-3]|</td><td>|[1-1]| + |[2-3]|</td></tr>
<tr><td></td><td></td><td></td><td></td><td></td><td>|[1-2]| + |[3-3]|</td></tr>
</table>

Range = 3

I am expecting to see such a Graph!!

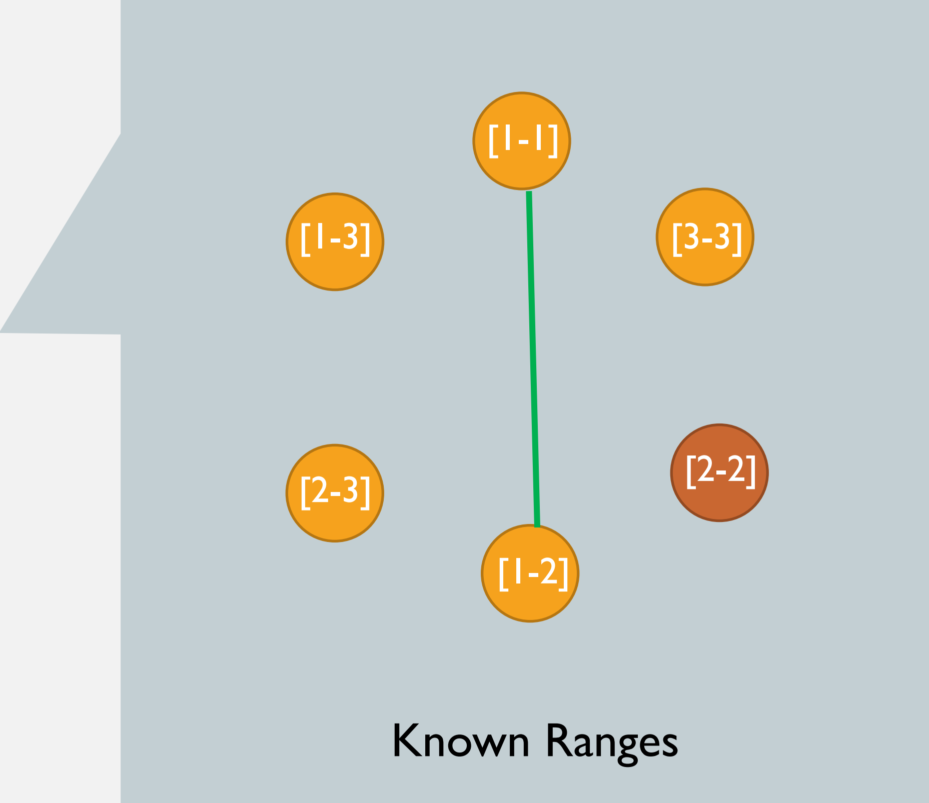


Known Ranges

|[1-1]|	|[2-2]|	|[3-3]|	|[1-2]|	|[2-3]|	|[1-3]|
			|[1-1]| + |[2-2]|	|[2-2]| + |[3-3]|	|[1-1]| + |[2-3]|
					|[1-2]| + |[3-3]|

Range = 3

I can connect [1-1] to [1-2] becase there is another node, i.e. [2-2] such that |[1-2]| = |[1-1]| + |[2-2]|

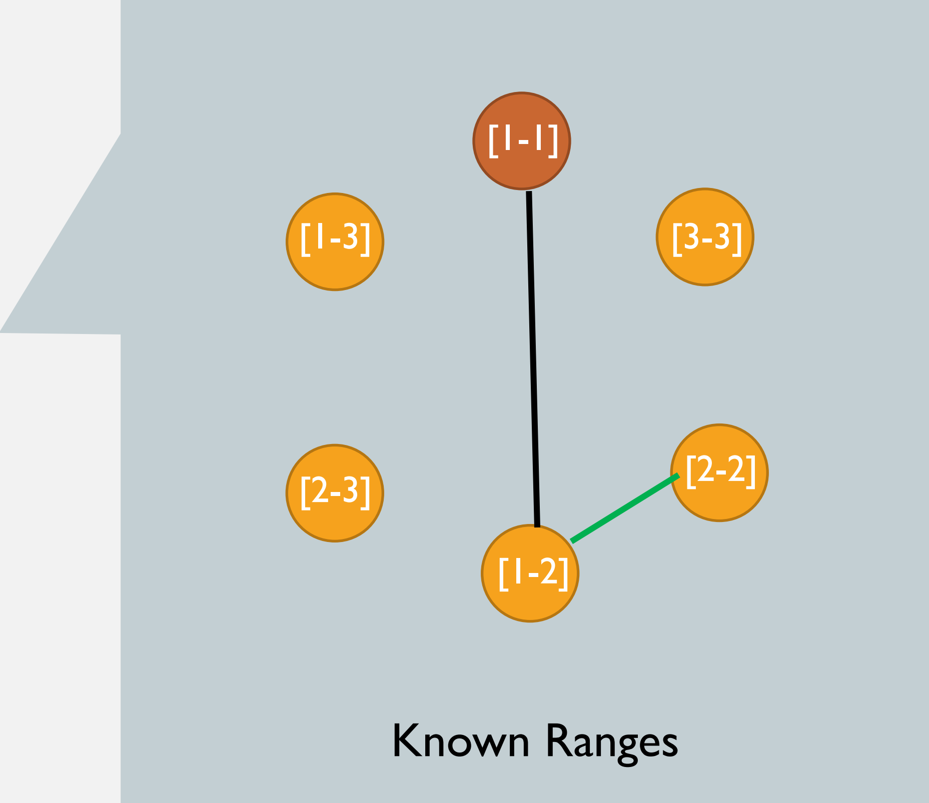


Known Ranges

|[1-1]|	|[2-2]|	|[3-3]|	|[1-2]|	|[2-3]|	|[1-3]|
			|[1-1]| + |[2-2]|	|[2-2]| + |[3-3]|	|[1-1]| + |[2-3]|
					|[1-2]| + |[3-3]|

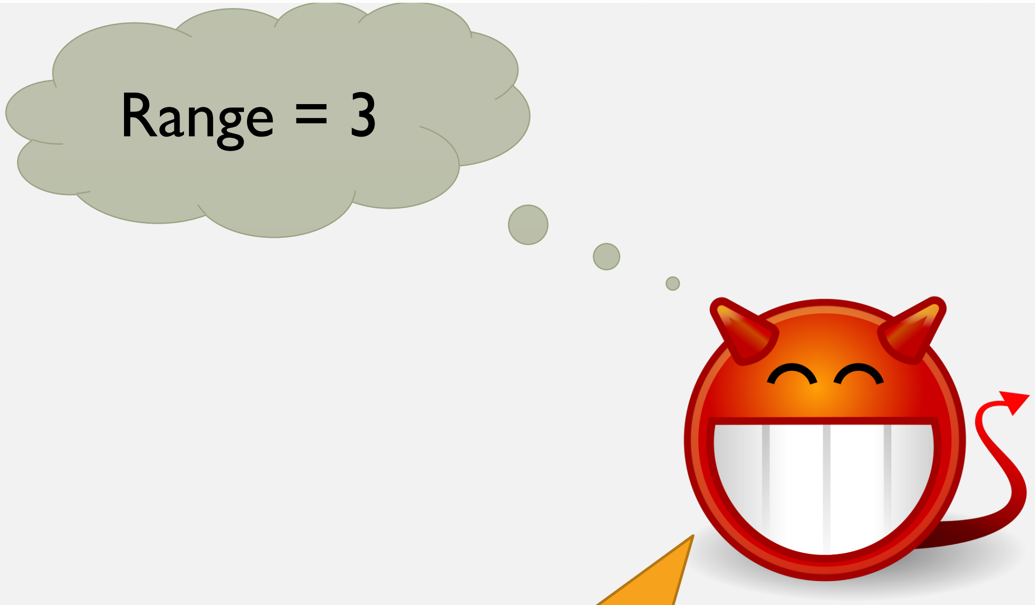
Range = 3

I can connect [2-2] to [1-2] becase there is another node, i.e. [1-1] such that |[1-2]| = |[1-1]| + |[2-2]|

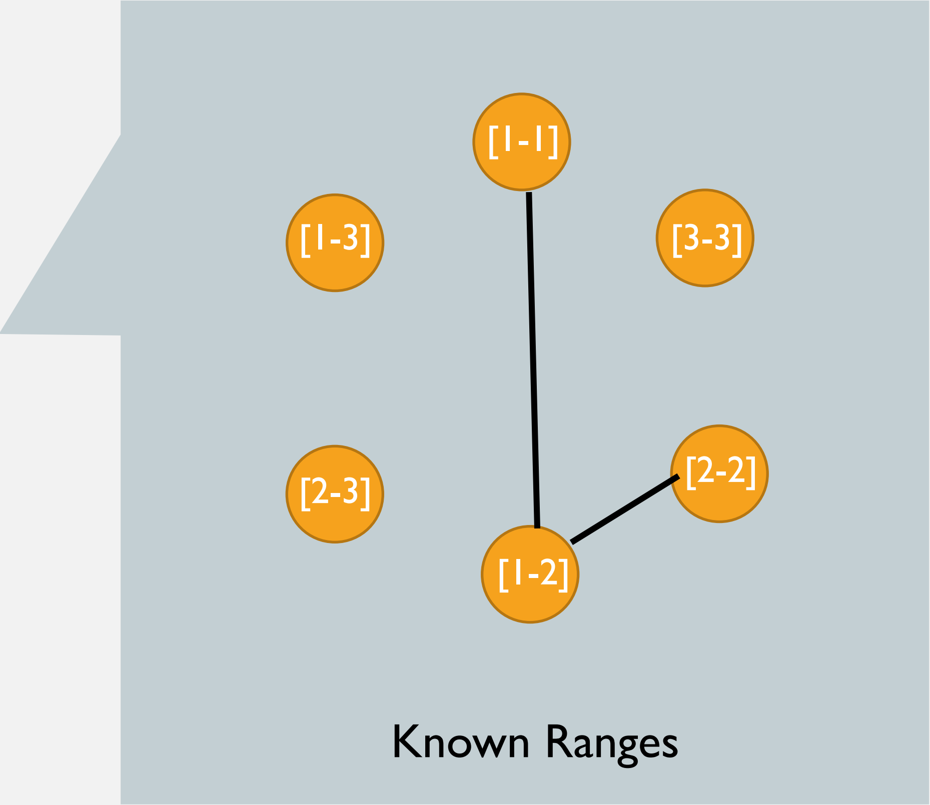


Known Ranges

|[1-1]|	|[2-2]|	|[3-3]|	|[1-2]|	|[2-3]|	|[1-3]|
			|[1-1]| + |[2-2]|	|[2-2]| + |[3-3]|	|[1-1]| + |[2-3]|
					|[1-2]| + |[3-3]|



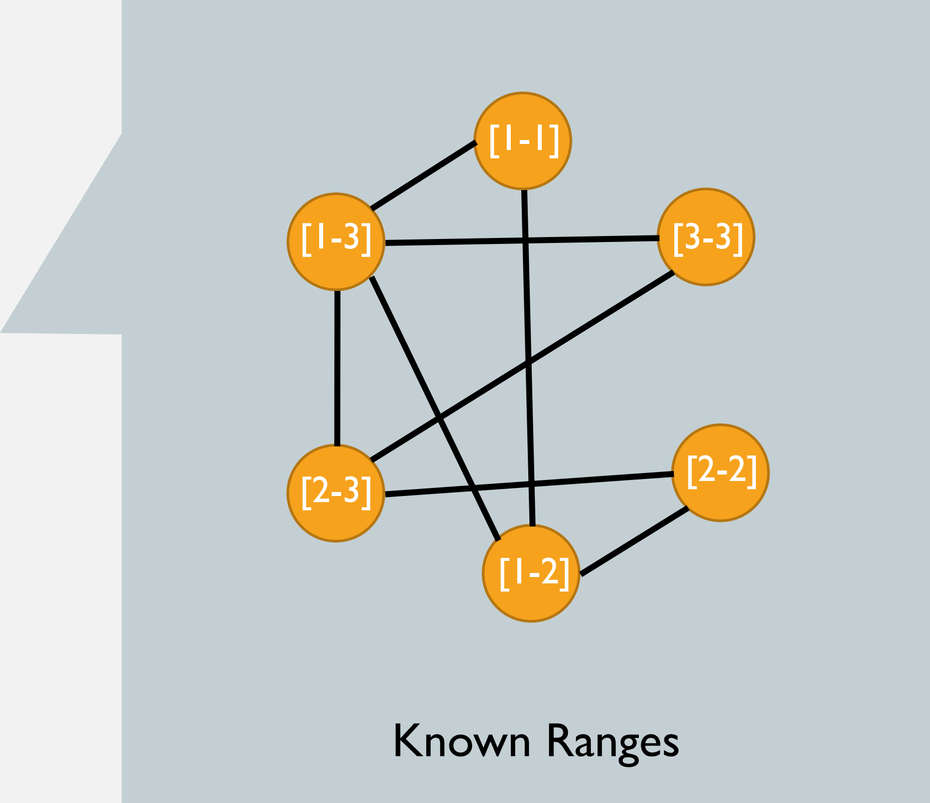
I can continue forming these connections …



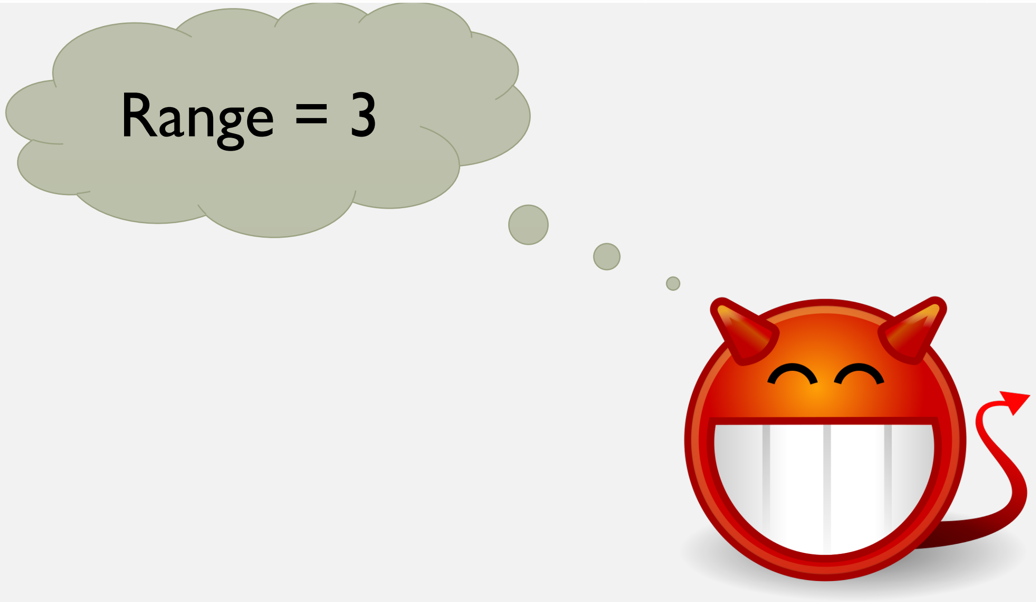
|[1-1]|	|[2-2]|	|[3-3]|	|[1-2]|	|[2-3]|	|[1-3]|
			|[1-1]| + |[2-2]|	|[2-2]| + |[3-3]|	|[1-1]| + |[2-3]|
					|[1-2]| + |[3-3]|

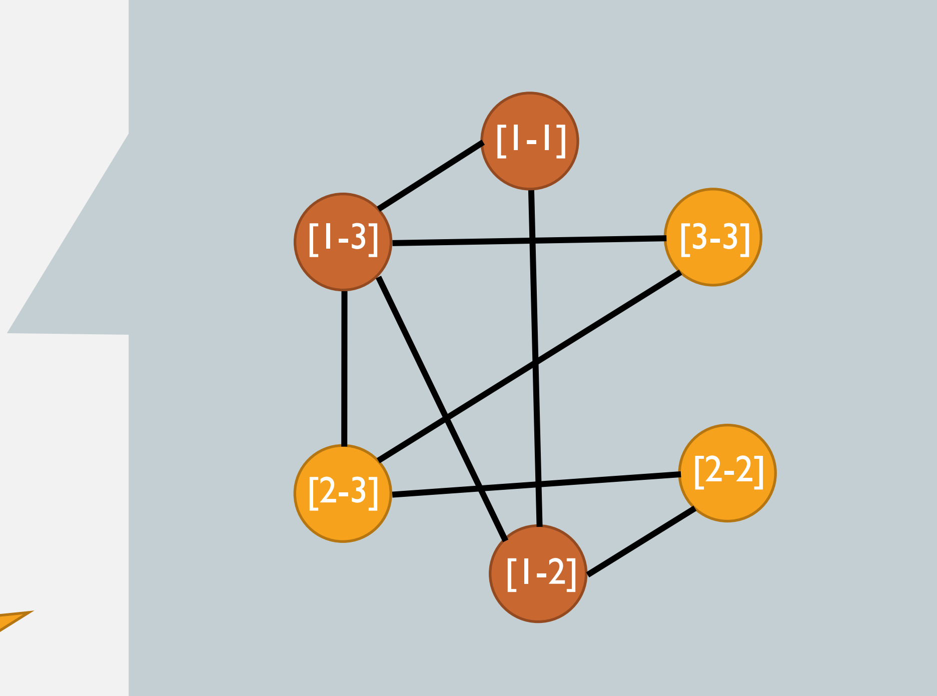
Range = 3

Done!



|[1-1]|	|[2-2]|	|[3-3]|	|[1-2]|	|[2-3]|	|[1-3]|
			|[1-1]| + |[2-2]|	|[2-2]| + |[3-3]|	|[1-1]| + |[2-3]|
					|[1-2]| + |[3-3]|





Known Ranges

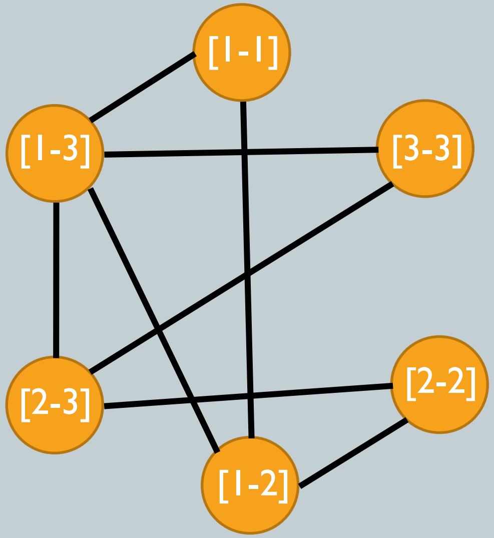
Claim: Nodes of the form [1-i] form a Clique!

|[1-1]|	**|[2-2]|**	**|[3-3]|**	**|[1-2]|**	**|[2-3]|**	**|[1-3]|**
			|[1-1]| + |[2-2]|	|[2-2]| + |[3-3]|	|[1-1]| + |[2-3]|
					|[1-2]| + |[3-3]|

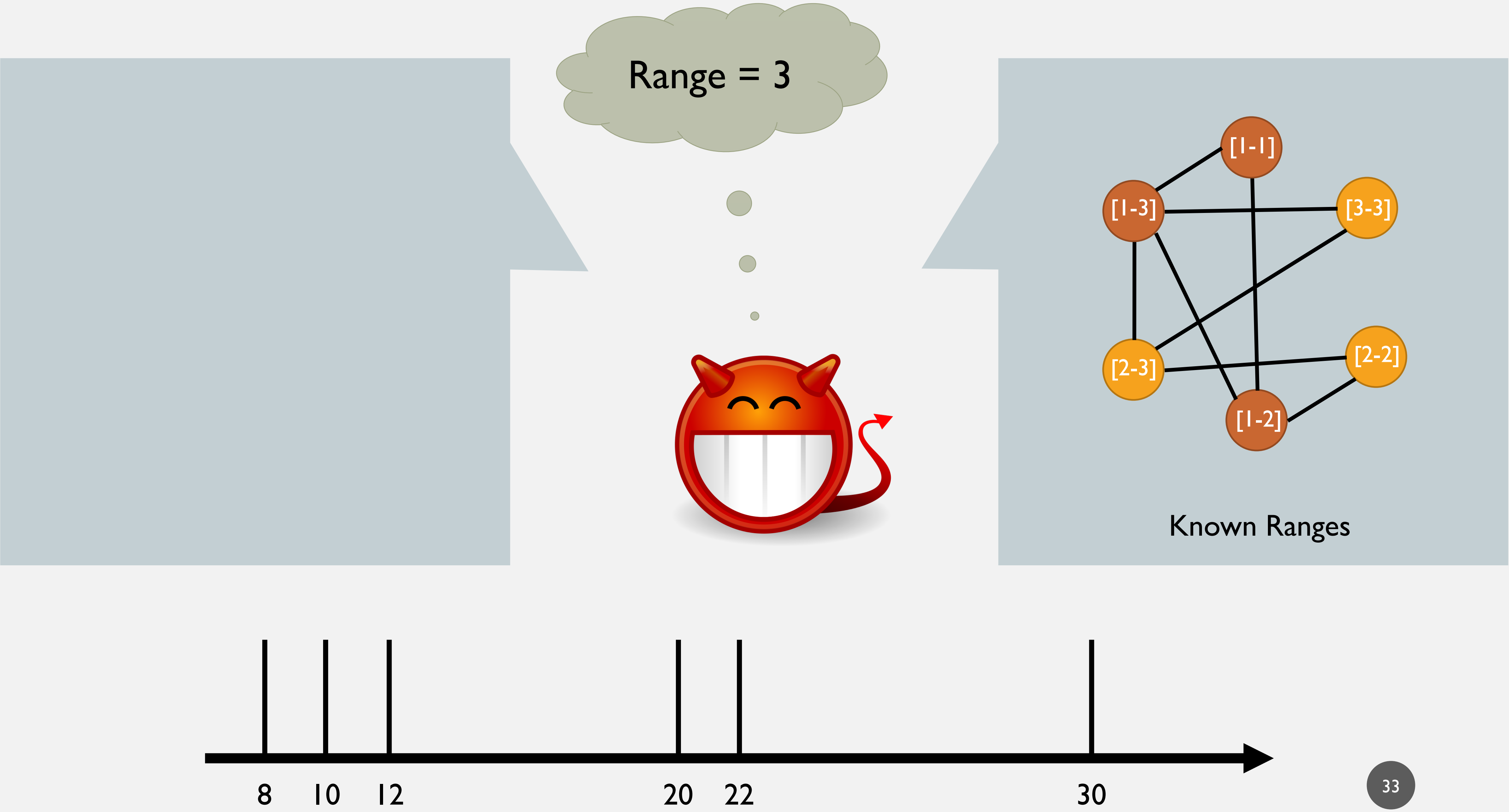
Range = 3

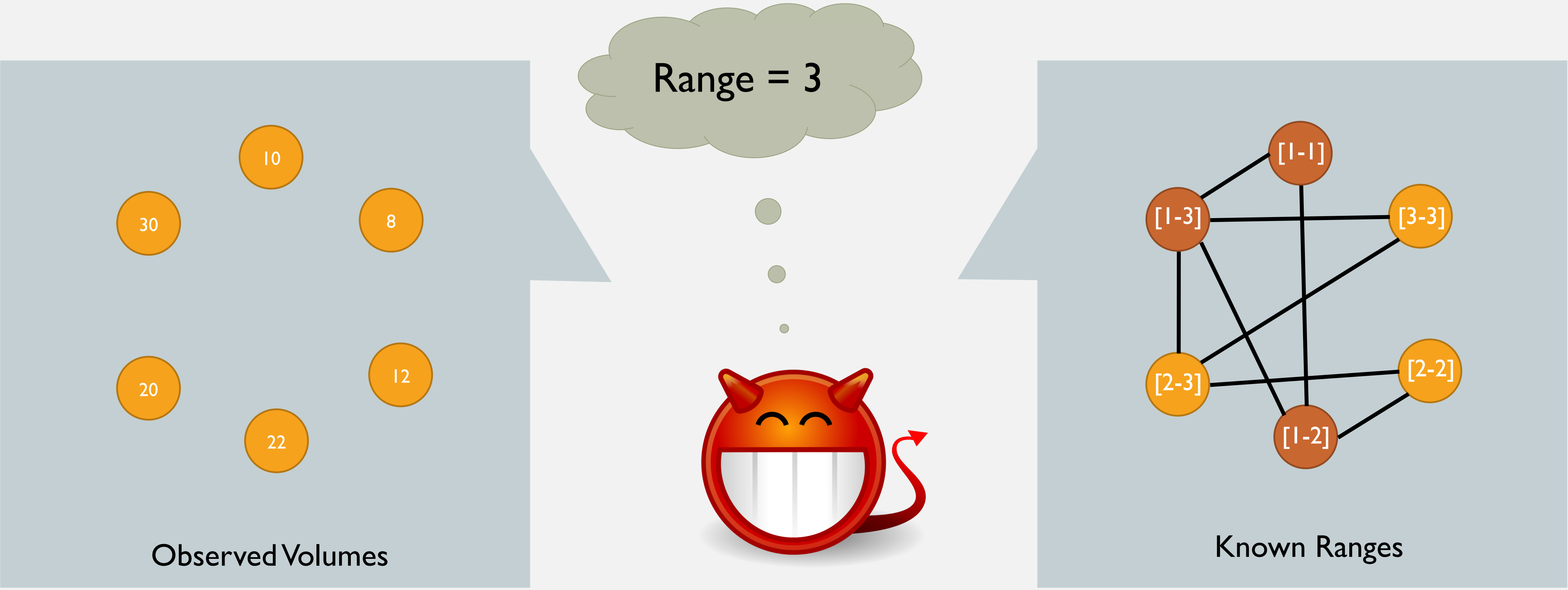
Proof Sketch.

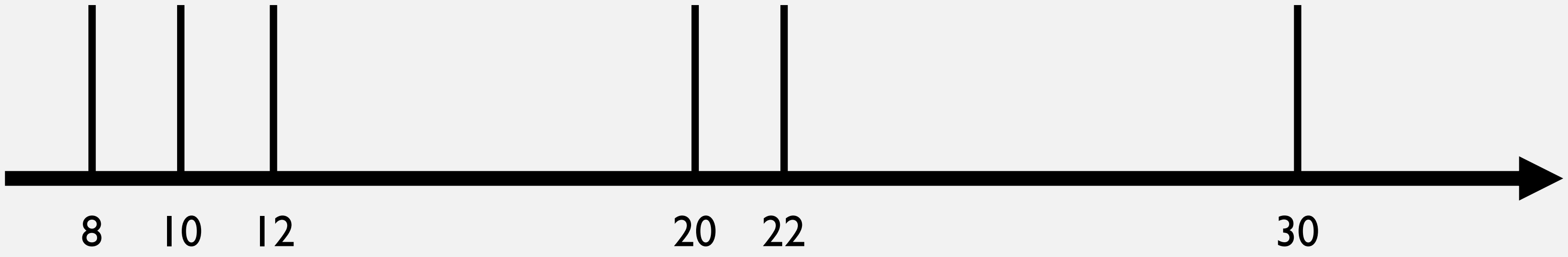
- Take two distinct nodes [1-i] and [1-j] (assume i < j)
- **Argument** : |[1-j]| = |[1-i]| + |[(i + 1)- j]|
  - Any database value between [1-j] is in [1-i] or [(i+1)-j]
- By our definition of the graph construction there is an edge between [1-i] and [1-j]



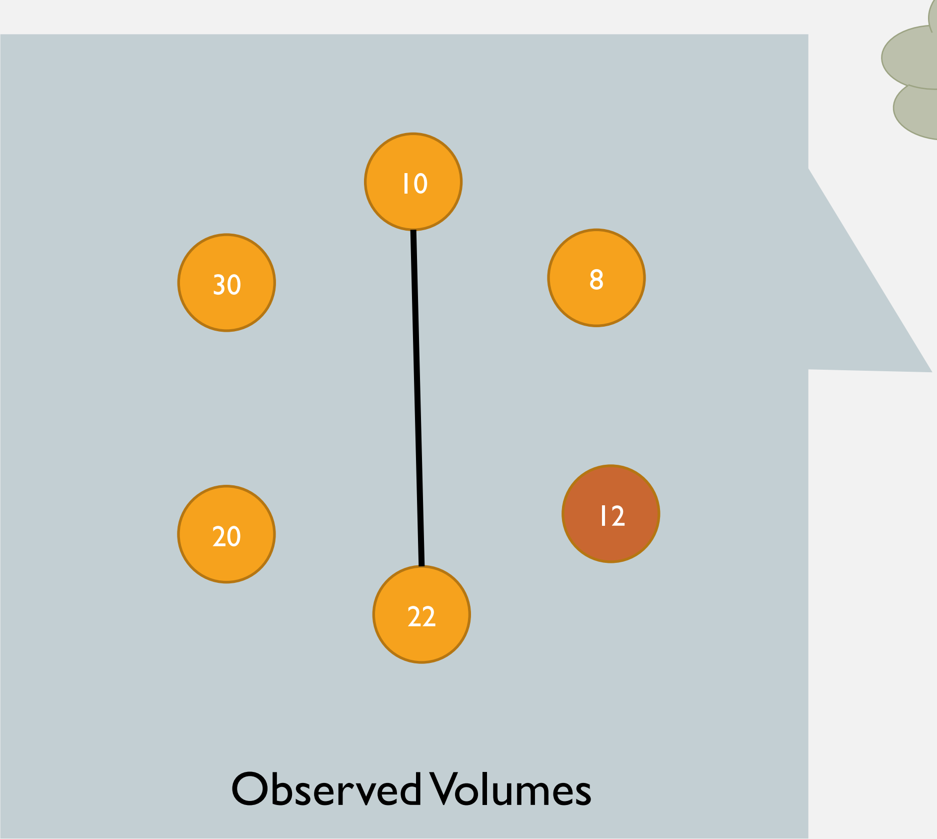
Known Ranges

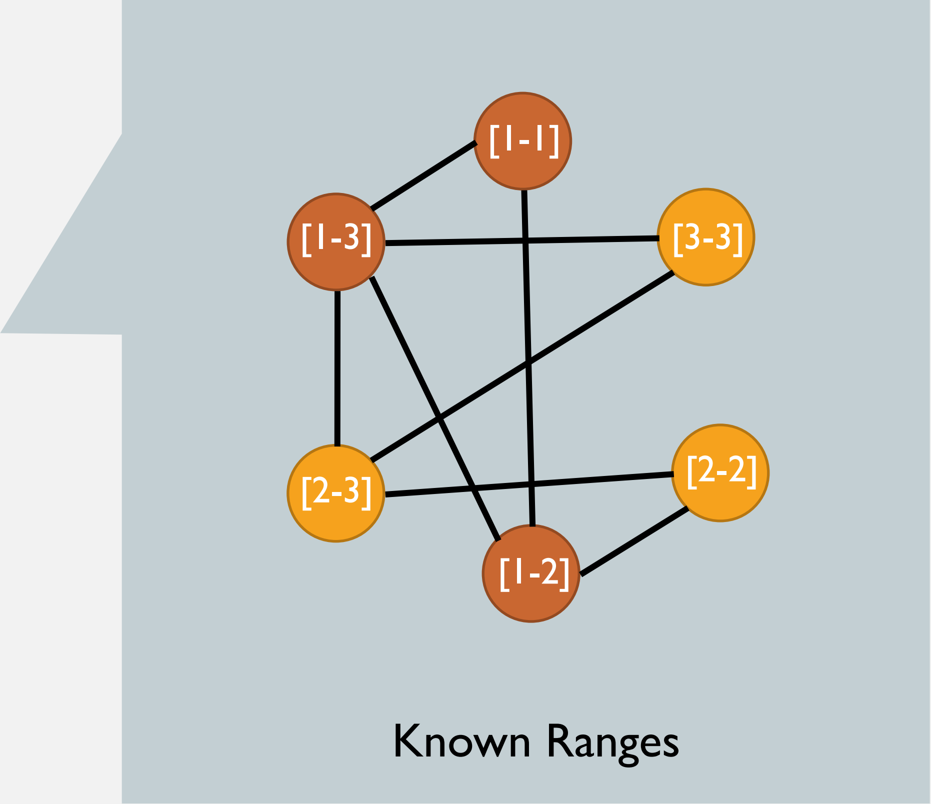

Range = 3
[1-1]
[1-3]
[3-3]
[2-3]
[2-2]
[1-2]
Known Ranges
8
10
12
20
22
30


Range = 3
10
30
8
20
12
22
Observed Volumes
[1-1]
[1-3]
[3-3]
[2-3]
[2-2]
[1-2]
Known Ranges


8
10
12
20
22
30

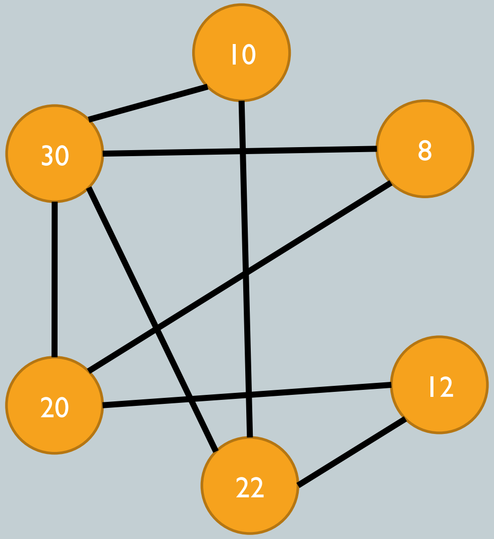
Range = 3



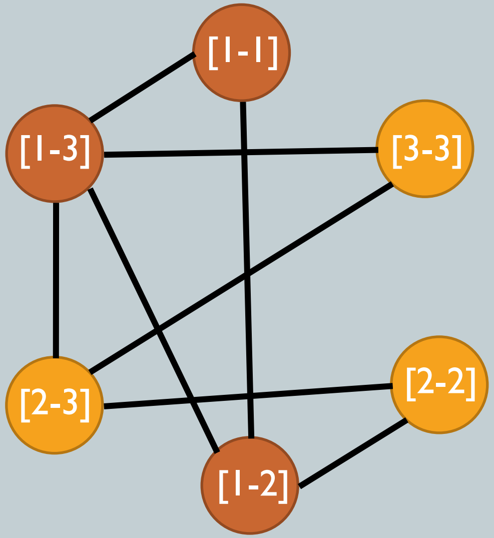


Let’s identify the connection between nodes …

Range = 3



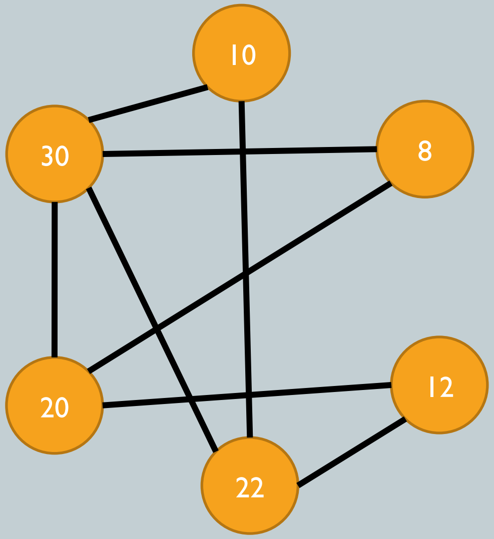
Observed Volumes



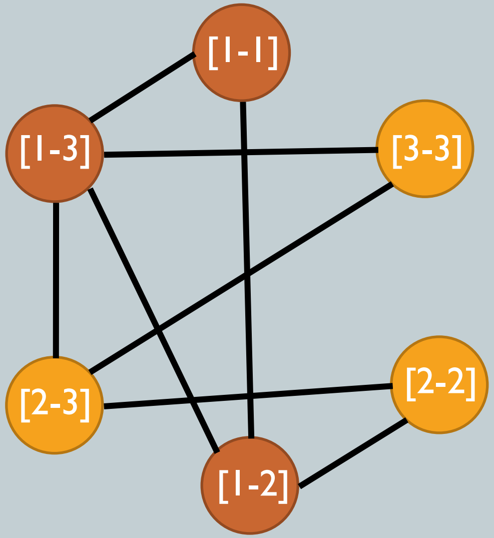
Known Ranges

Done!

Range = 3



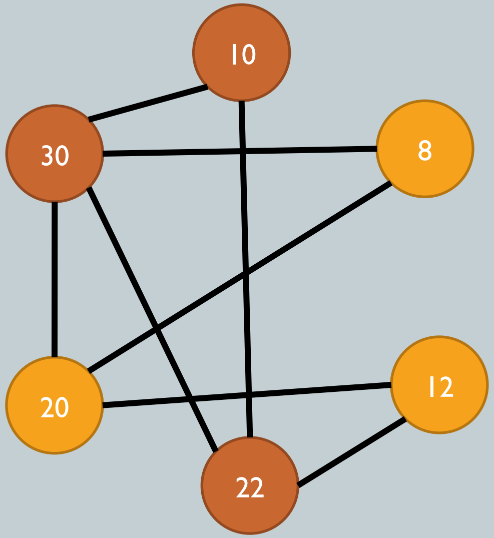
Observed Volumes



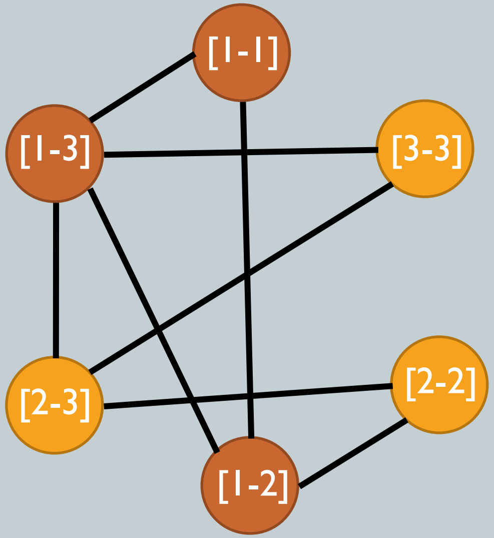
Known Ranges

Where is the Clique???

Range = 3

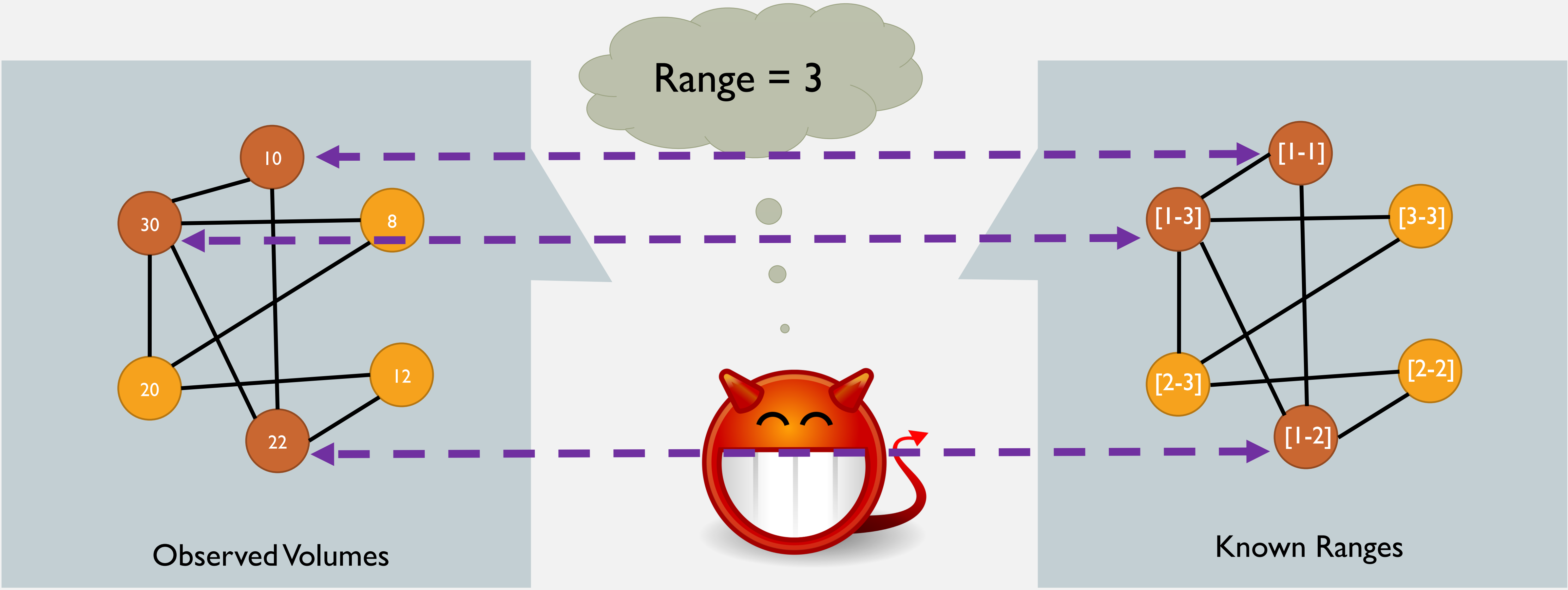


Observed Volumes

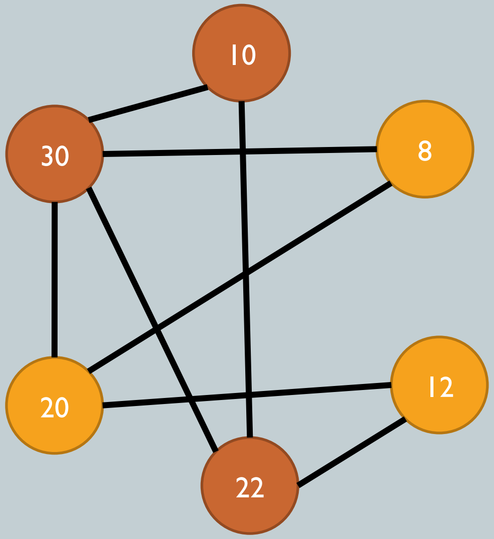


Known Ranges

Where is the Clique???

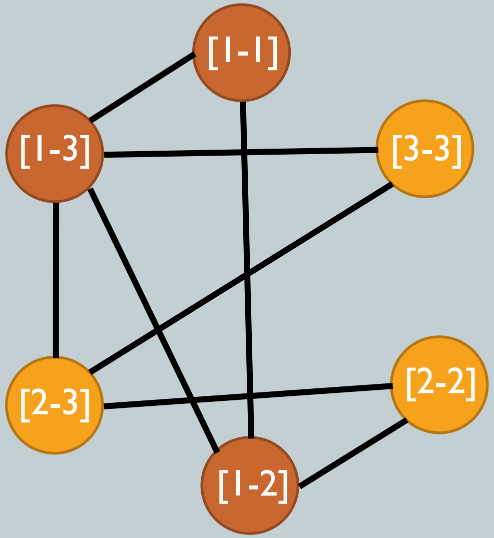


Claim: Nodes of the form [1-i] form a Clique!



Observed Volumes

Range = 3



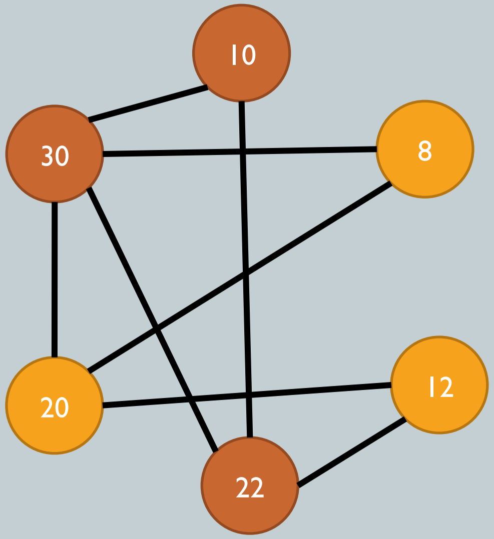
Known Ranges

$|[1\text{-}1]| = 10$

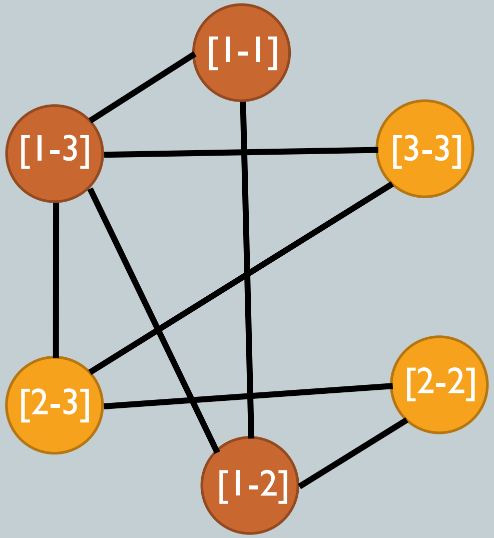
$|[1\text{-}2]| = 22$

$|[1\text{-}3]| = 30$

Range = 3



Observed Volumes



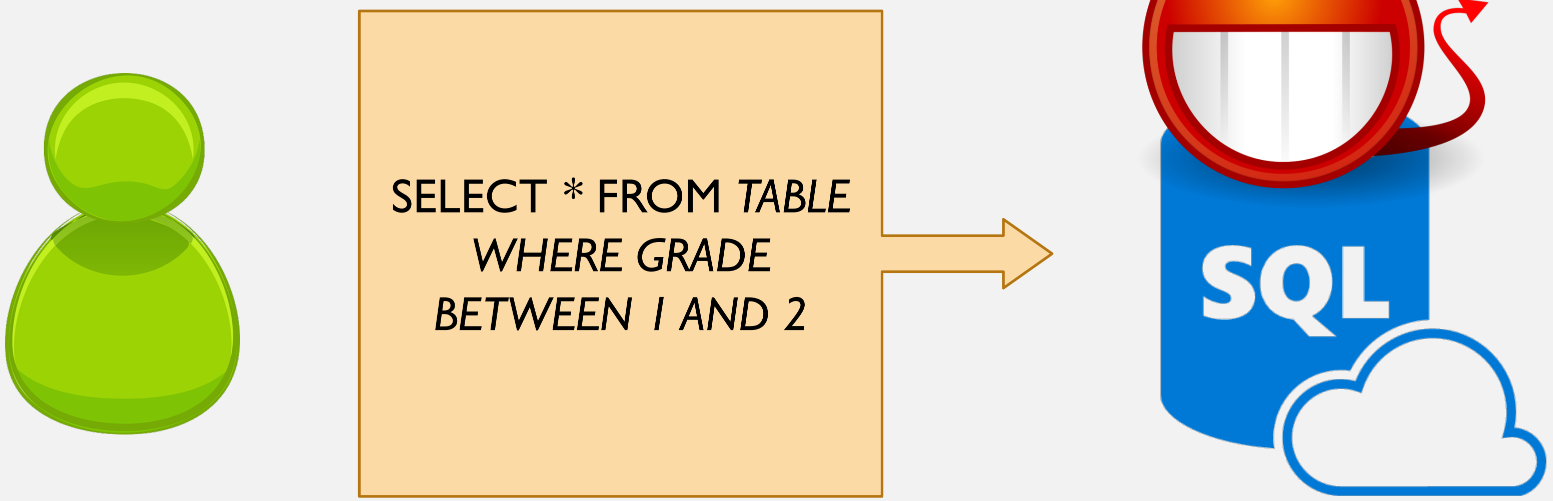
Known Ranges

|[1-1]| = 10

|[1-2]| = 22 → **|[2-2]| = 12**

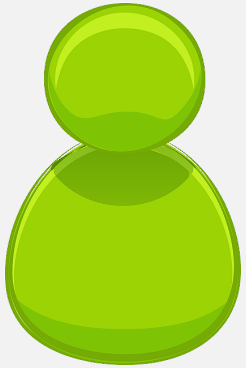
|[1-3]| = 30 → **|[3-3]| = 8**

This is a range query on a column which has a range N



SELECT * FROM *TABLE*
WHERE GRADE
BETWEEN 1 AND 2

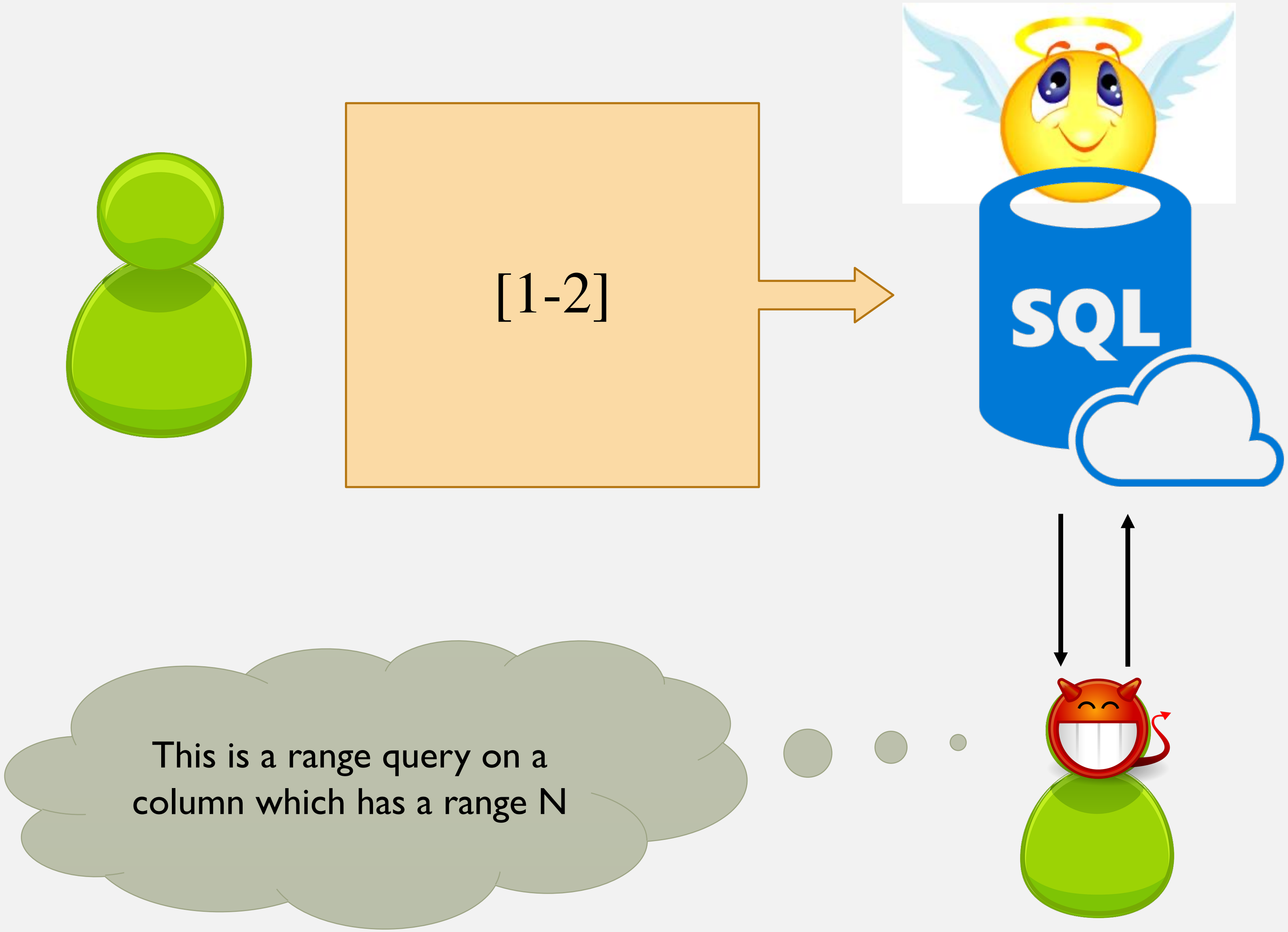
Name	Grade	
Edgar	1	10
…	1	
Jack	1	
Casey	2	12
…	2	
Nina	2	
Dennis	3	8
…	3	
Paige	3	



SELECT * FROM *TABLE*
WHERE GRADE
BETWEEN 1 AND 2

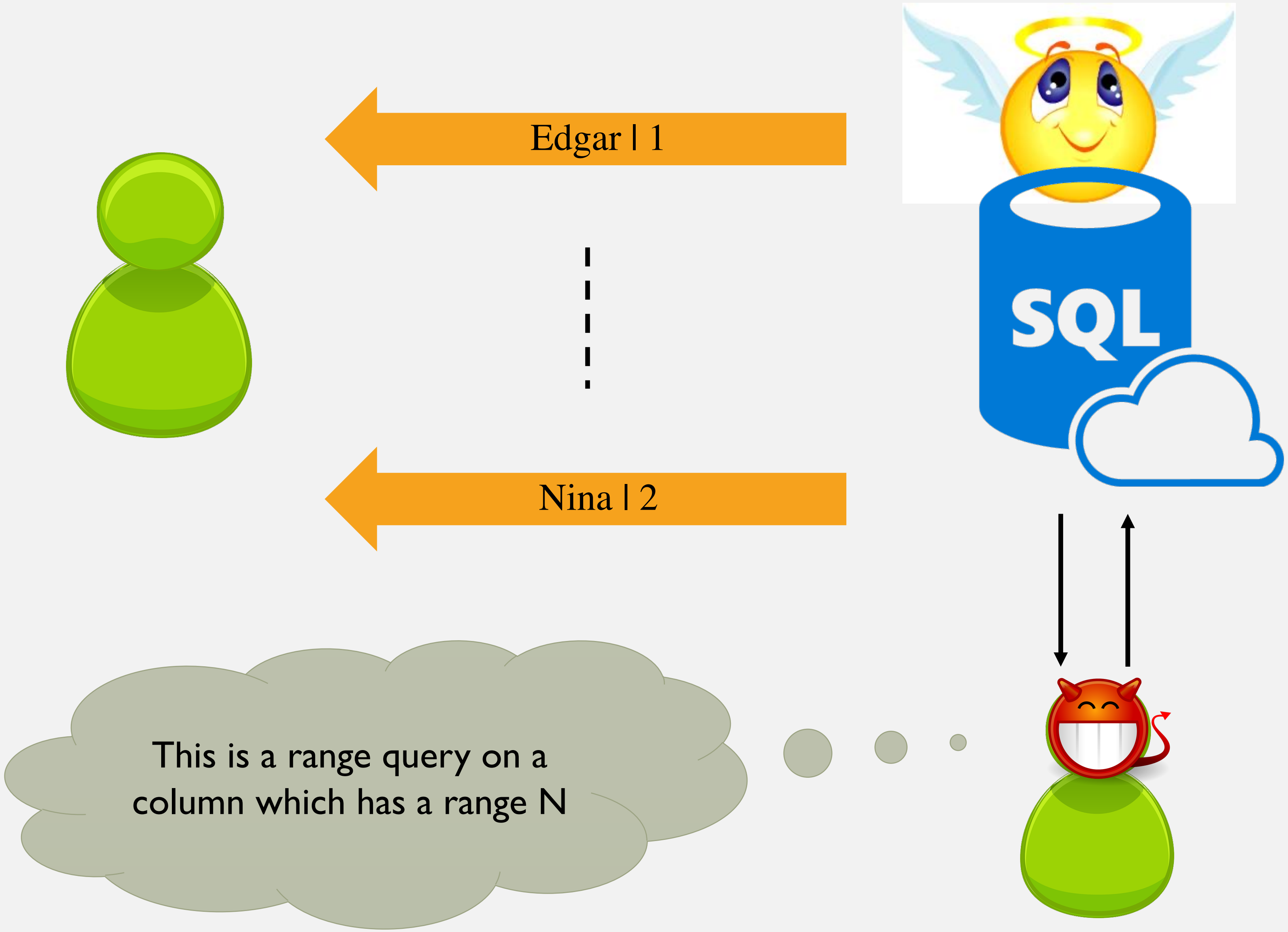
Name	Grade	
Edgar	1	10
…	1	
Jack	1	
Casey	2	12
…	2	
Nina	2	
Dennis	3	8
…	3	
Paige	3	

This is a range query on a
column which has a range N



This is a range query on a column which has a range N

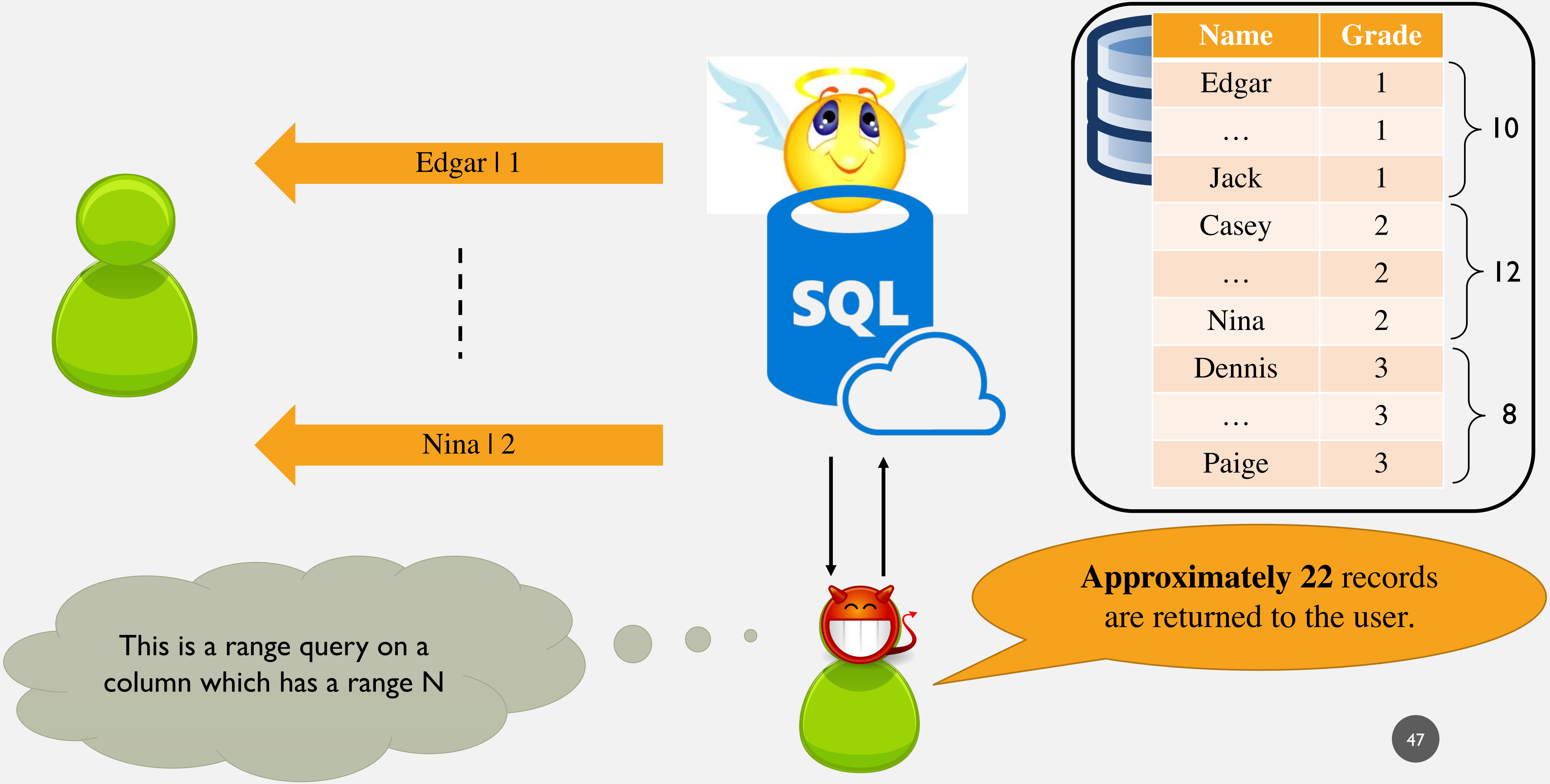
Name	Grade	
Edgar	1	10
…	1	
Jack	1	
Casey	2	12
…	2	
Nina	2	
Dennis	3	8
…	3	
Paige	3	


Edgar | 1

Nina | 2

This is a range query on a column which has a range N

Name	Grade	
Edgar	1	10
…	1	
Jack	1	
Casey	2	12
…	2	
Nina	2	
Dennis	3	8
…	3	
Paige	3	

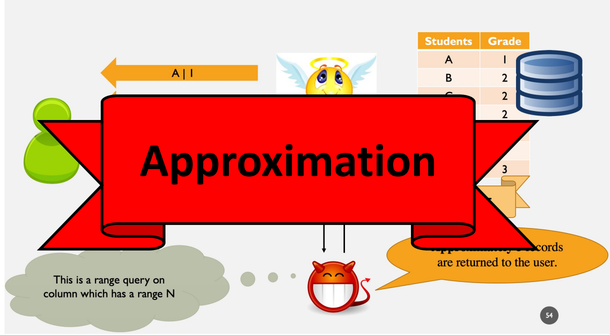


Name	Grade
Edgar	1
…	1
Jack	1
Casey	2
…	2
Nina	2
Dennis	3
…	3
Paige	3

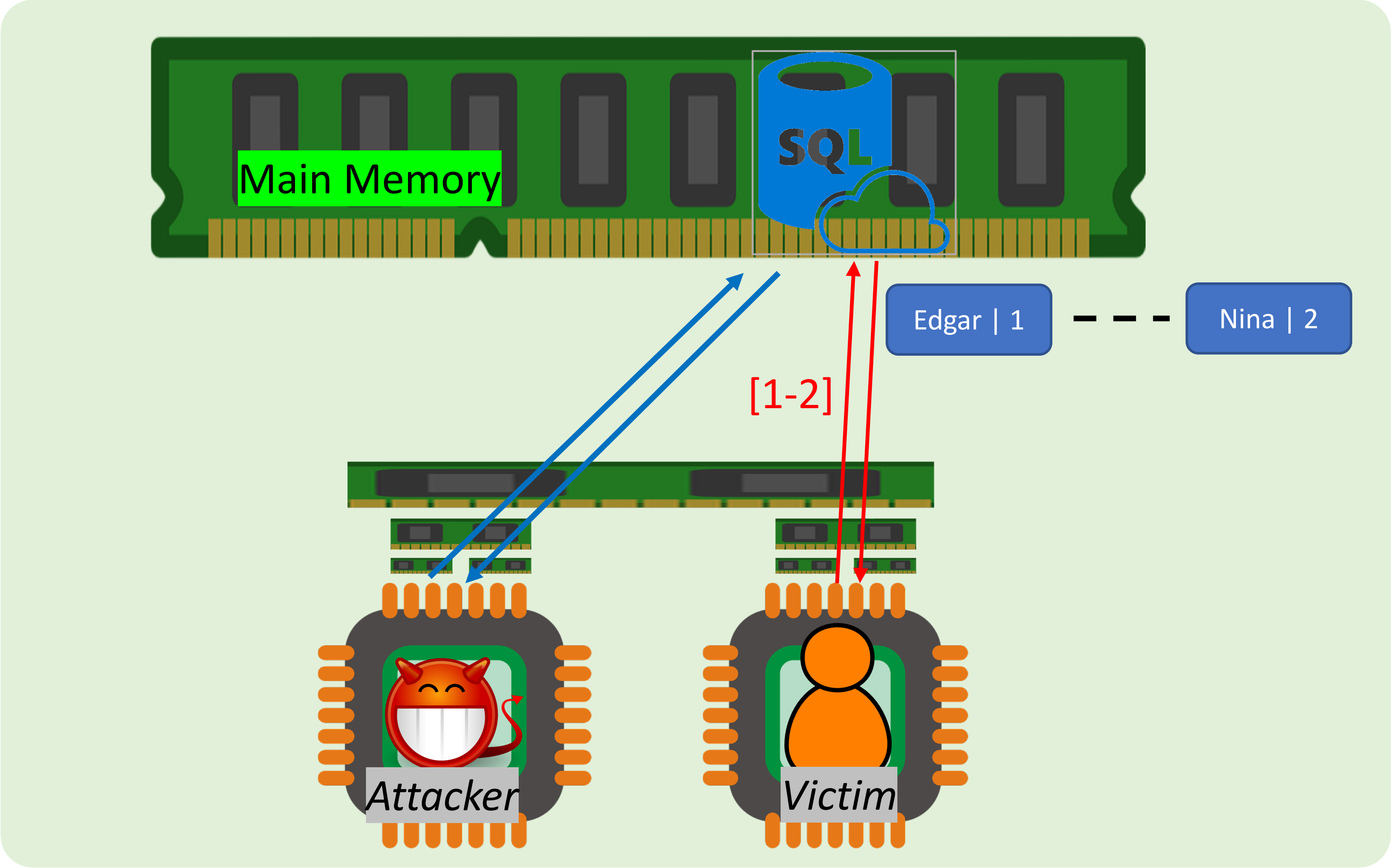
Approximately 22 records are returned to the user.

This is a range query on a column which has a range N

Cache Attack Model

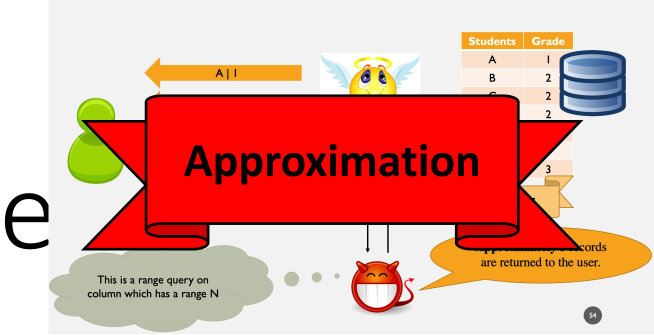


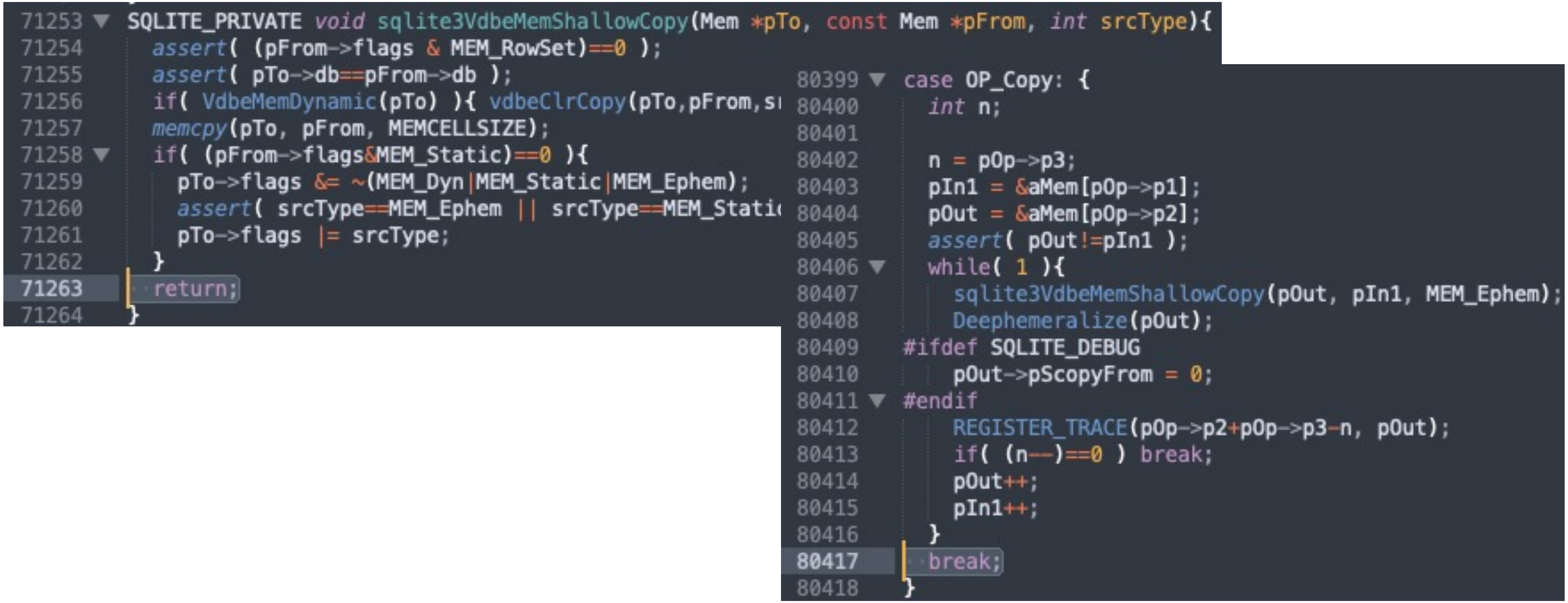
Approximation
A | I
Students
Grade
This is a range query on column which has a range N
are returned to the user.



Name	Grade	
Edgar	1	10
…	1	
Jack	1	
Casey	2	12
…	2	
Nina	2	
Dennis	3	8
…	3	
Paige	3	

The Lines That Correspond to Volume


A | I
Students
Grade
A
1
B
2
2
2
3
Approximation
This is a range query on column which has a range N
are returned to the user.
54

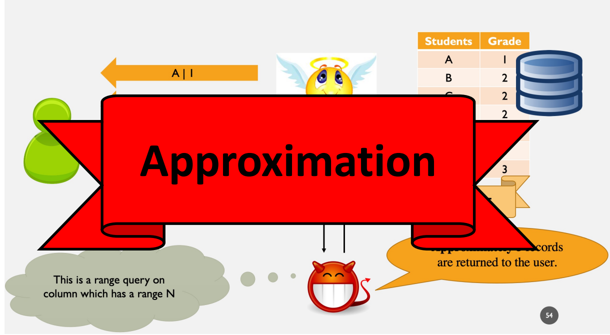


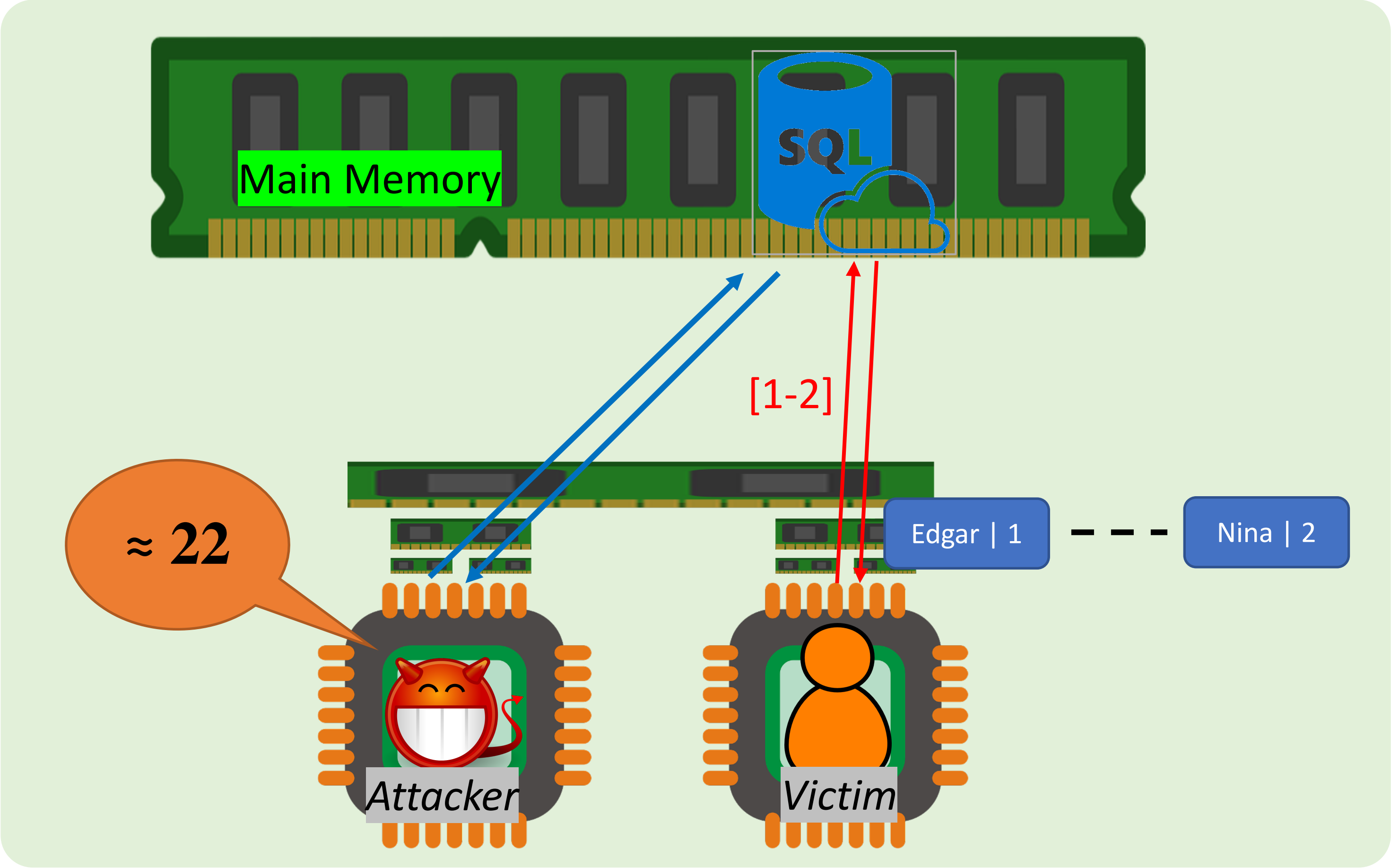
```
SQLITE_PRIVATE void sqlite3VdbeMemShallowCopy(Mem *pTo, const Mem *pFrom, int srcType){
  assert( (pFrom->flags & MEM_RowSet)==0 );
  assert( pTo->db==pFrom->db );
  if( VdbeMemDynamic(pTo) ){ vdbeClrCopy(pTo,pFrom,s
  memcpy(pTo, pFrom, MEMCELLSIZE);
  if( (pFrom->flags&MEM_Static)==0 ){
    pTo->flags &= ~(MEM_Dyn|MEM_Static|MEM_Ephem);
    assert( srcType==MEM_Ephem || srcType==MEM_Stati
    pTo->flags |= srcType;
  }
  return;
}
```

```
case OP_Copy: {
  int n;

  n = pOp->p3;
  pIn1 = &aMem[pOp->p1];
  pOut = &aMem[pOp->p2];
  assert( pOut!=pIn1 );
  while( 1 ){
    sqlite3VdbeMemShallowCopy(pOut, pIn1, MEM_Ephem);
    Deephemeralize(pOut);
#ifdef SQLITE_DEBUG
    pOut->pScopyFrom = 0;
#endif
    REGISTER_TRACE(pOp->p2+pOp->p3-n, pOut);
    if( (n--)==0 ) break;
    pOut++;
    pIn1++;
  }
  break;
}
```

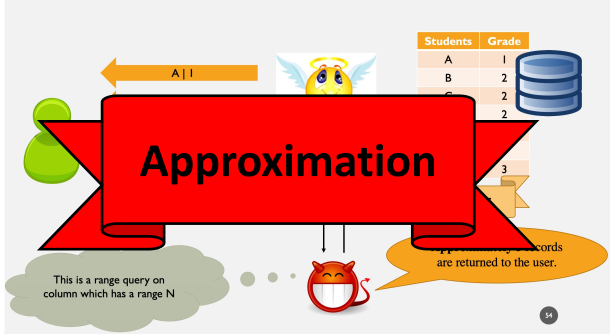
Cache Attack Model


Approximation
Students
Grade
A | I
This is a range query on column which has a range N
are returned to the user.



Name	Grade	
Edgar	1	10
…	1	
Jack	1	
Casey	2	12
…	2	
Nina	2	
Dennis	3	8
…	3	
Paige	3	

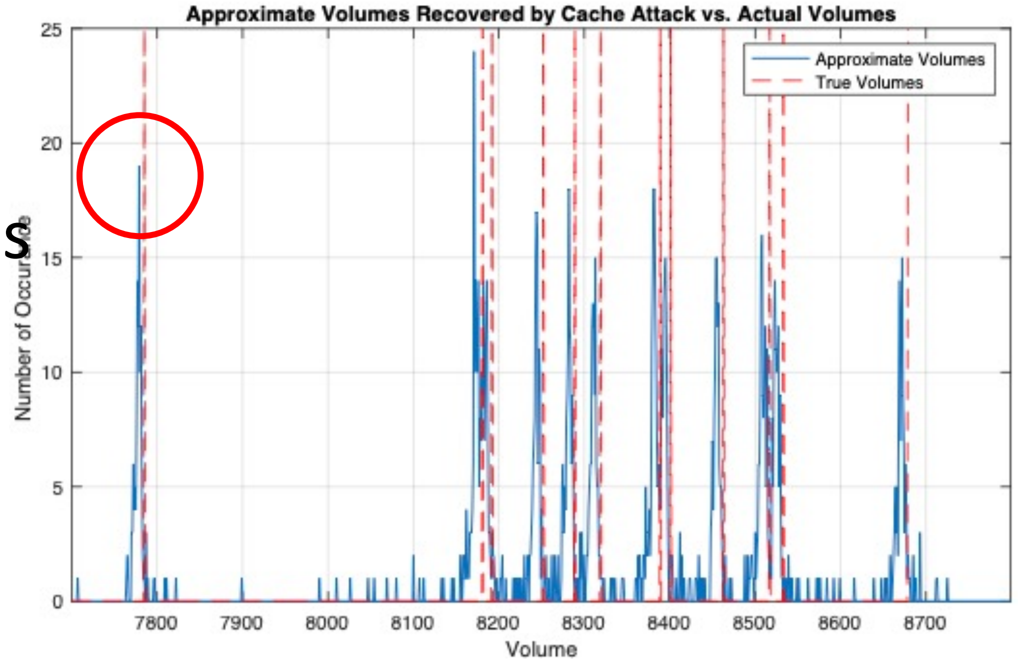
Flush ✓ Reload ✗



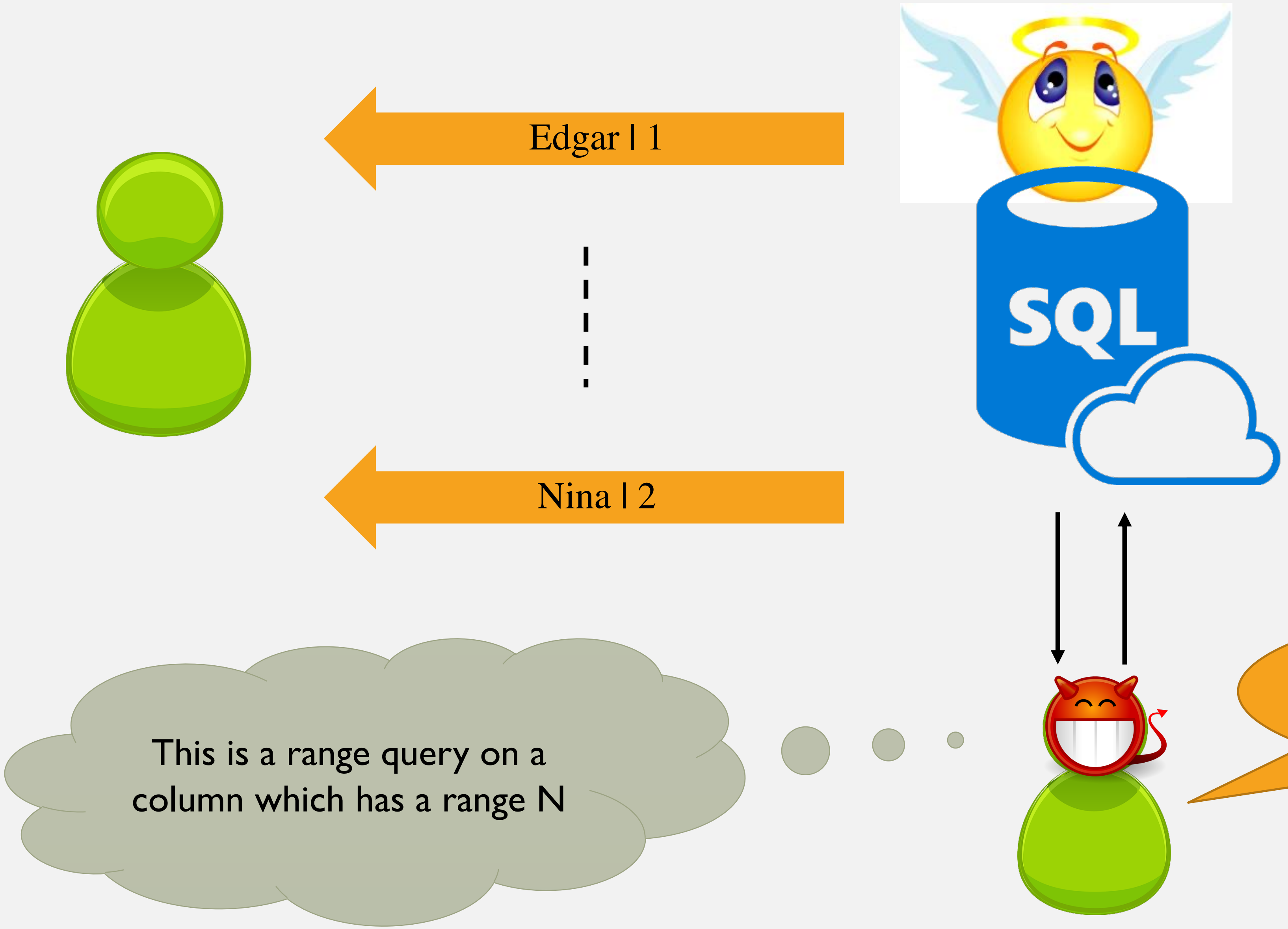
Students
Grade
A
B
A | I
Approximation
This is a range query on column which has a range N
are returned to the user.

Recovered Volumes

- Obtain an (Approximation of) Volume for each query
- Repeat the attack and aggregate the obtained volumes
- The **red dotted lines** are the exact volumes
- The **blue** line is the volumes obtained from cache attack
- They are quite <u>*close*</u> but not <u>*exactly*</u>



Peaks represent the Volumes

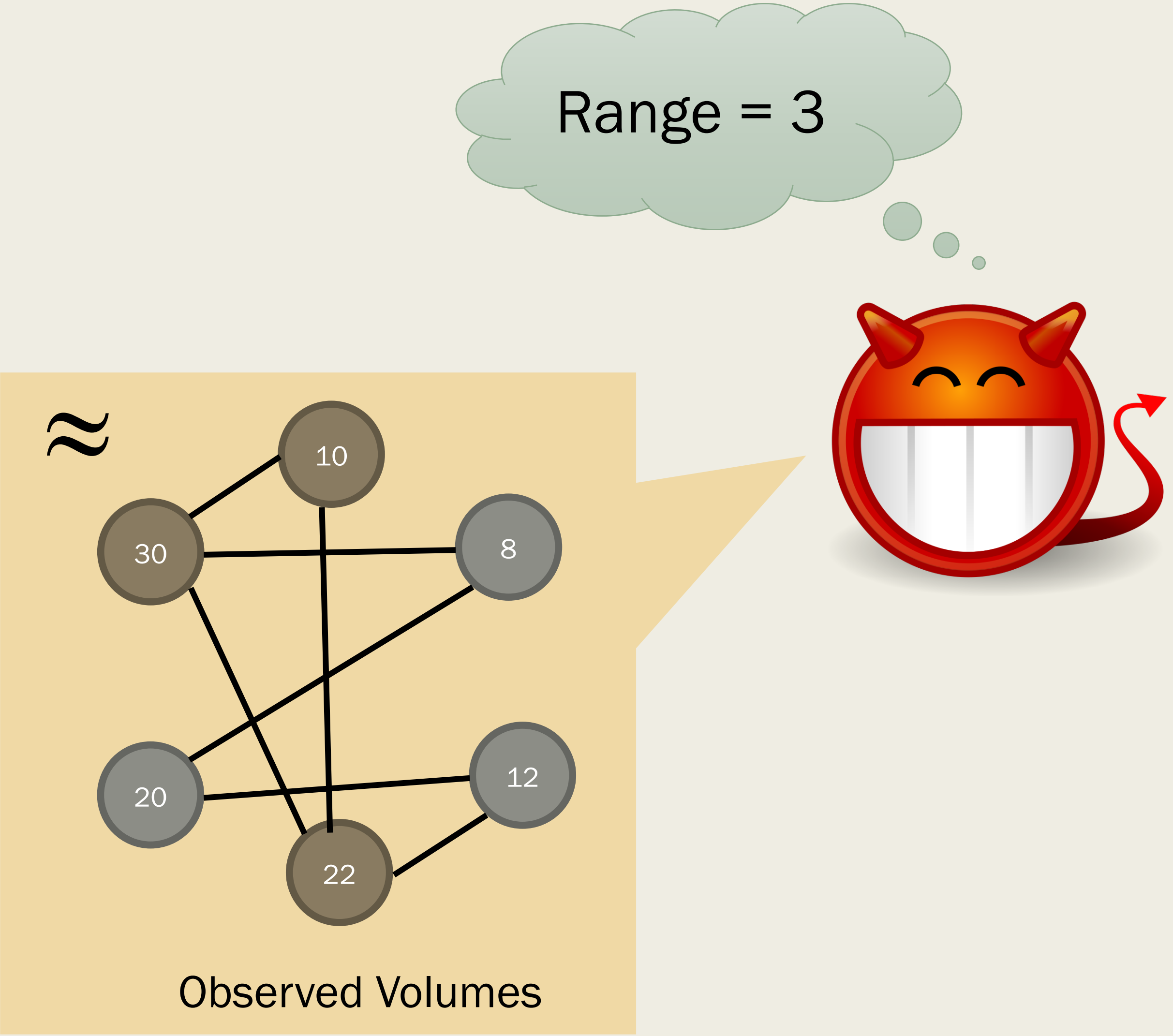


This is a range query on a column which has a range N

Name	Grade	
Edgar	1	
…	1	10
Jack	1	
Casey	2	
…	2	12
Nina	2	
Dennis	3	
…	3	8
Paige	3	

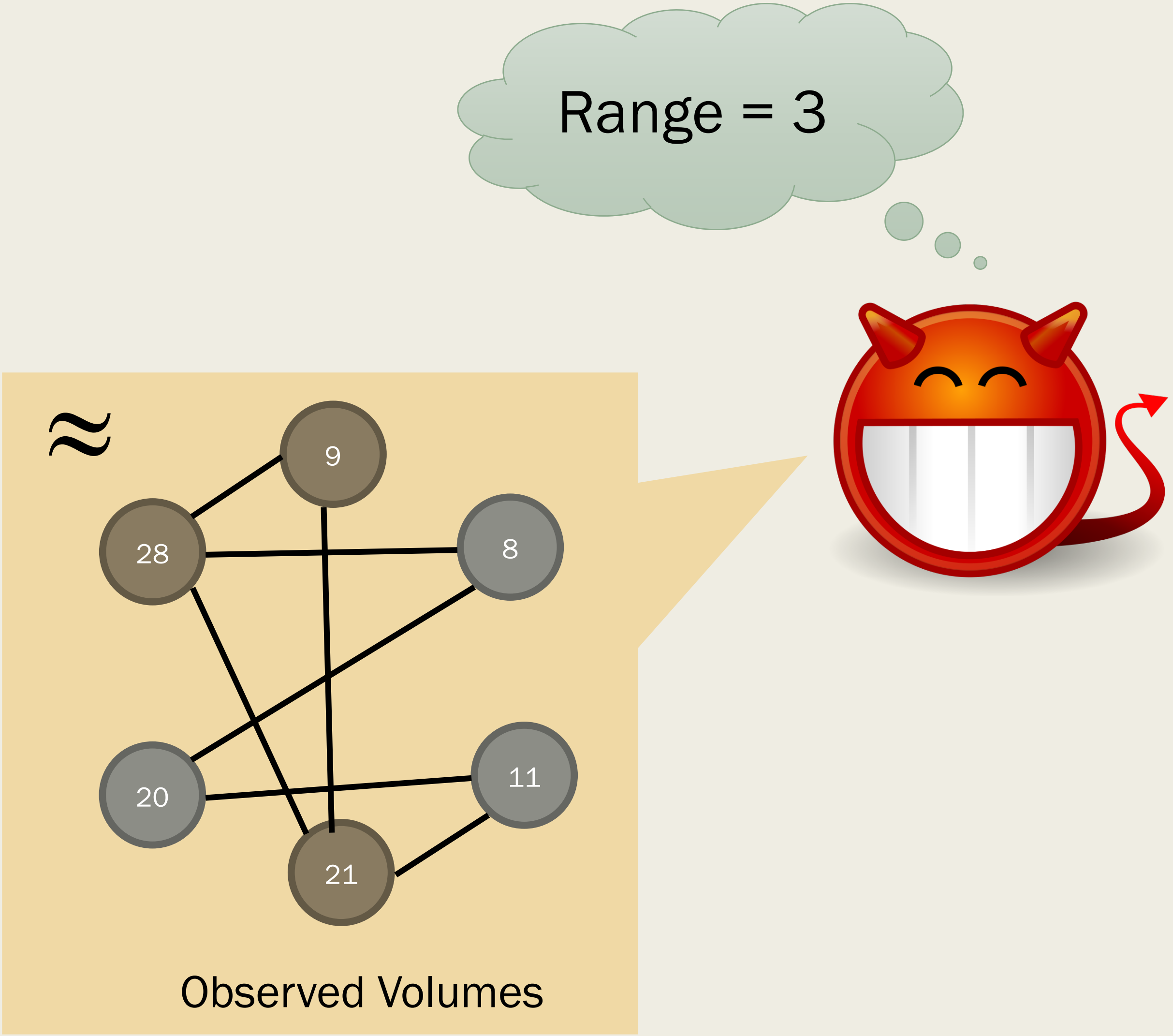
Approximately 22 records are returned to the user.

Approximate Volume and Graph Construction


Range = 3

Observed Volumes

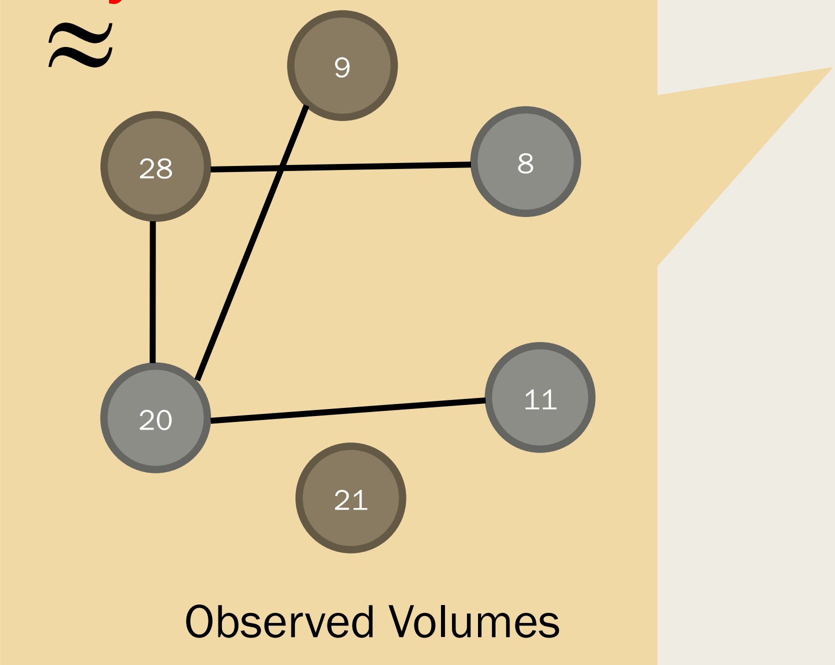
Approximate Volume and Graph Construction



Approximate Volume and Graph Construction

Range = 3

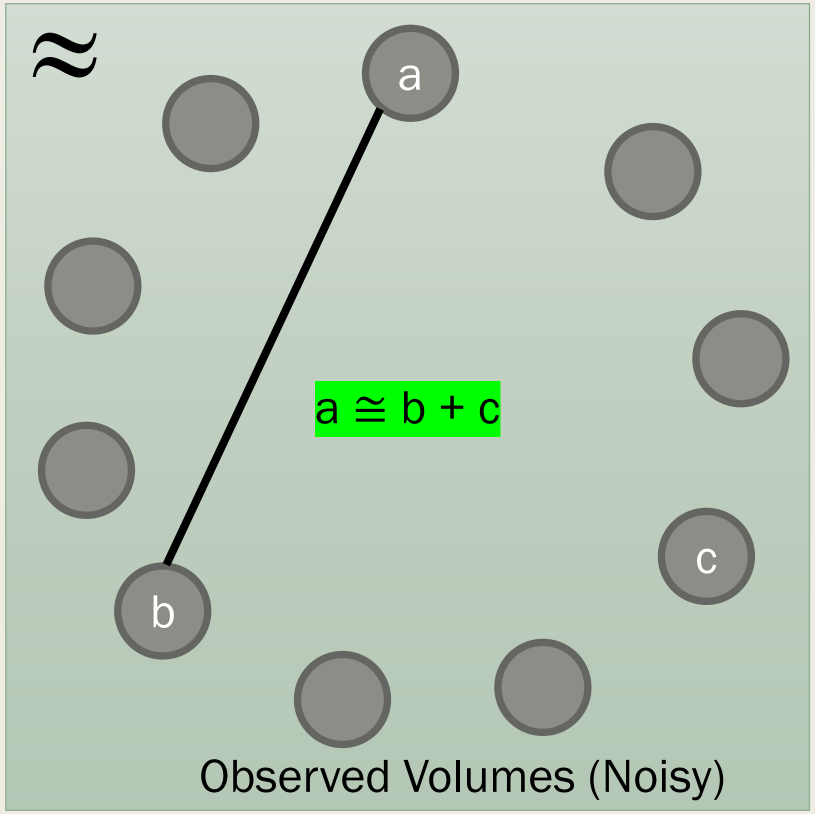
- Some Connections in the Graph might be missing
- The Clique Might not form properly
- We still can recover the (approximation of) database
  1. *Change the way we connect nodes*
  2. *Extend the Clique Finding Algorithm*



Observed Volumes

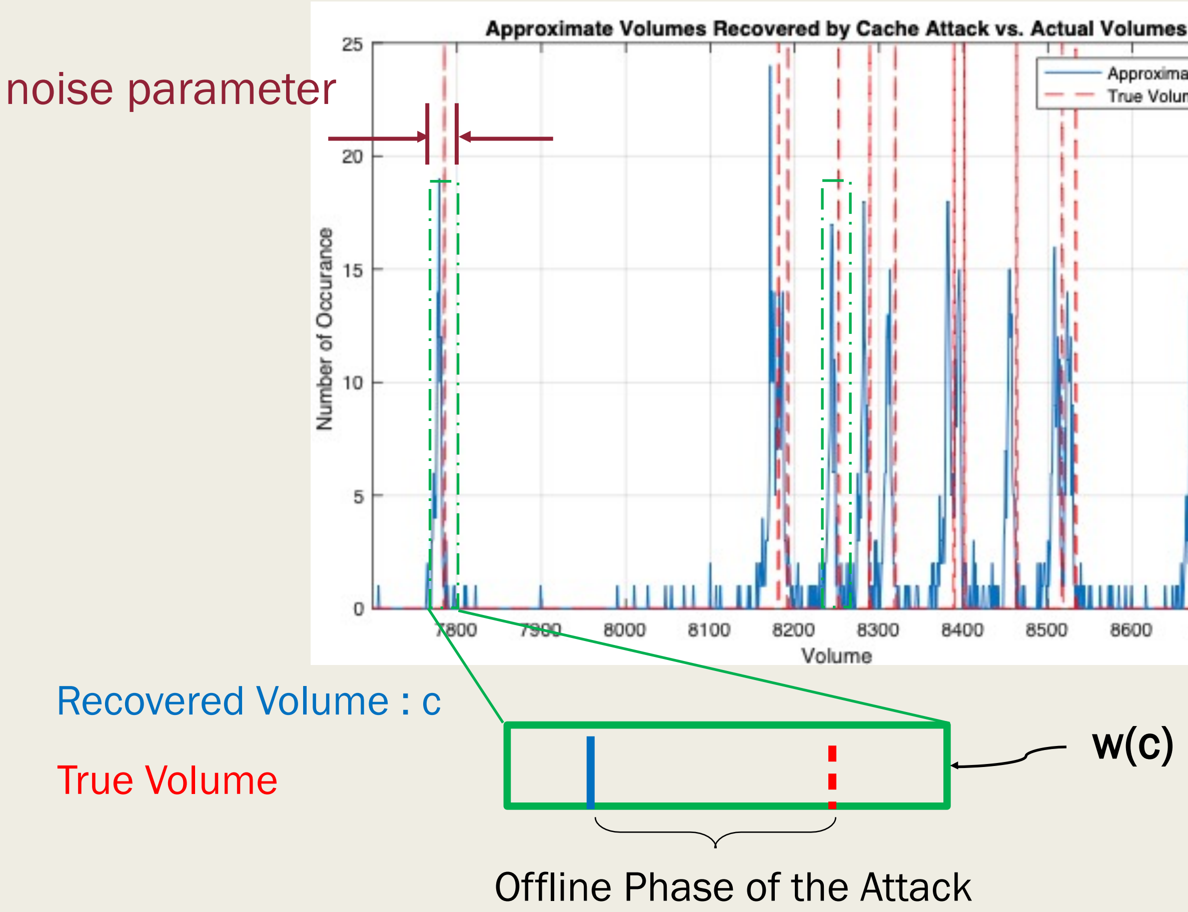
1. Constructing The Graph

- ≅ : is determined by <u>*noise parameter*</u>
  - *Obtained in a preprocessing step which involves mounting the attack on a database known to the attacker.*



1. Constructing The Graph

- ≅ : is determined by *noise parameter*



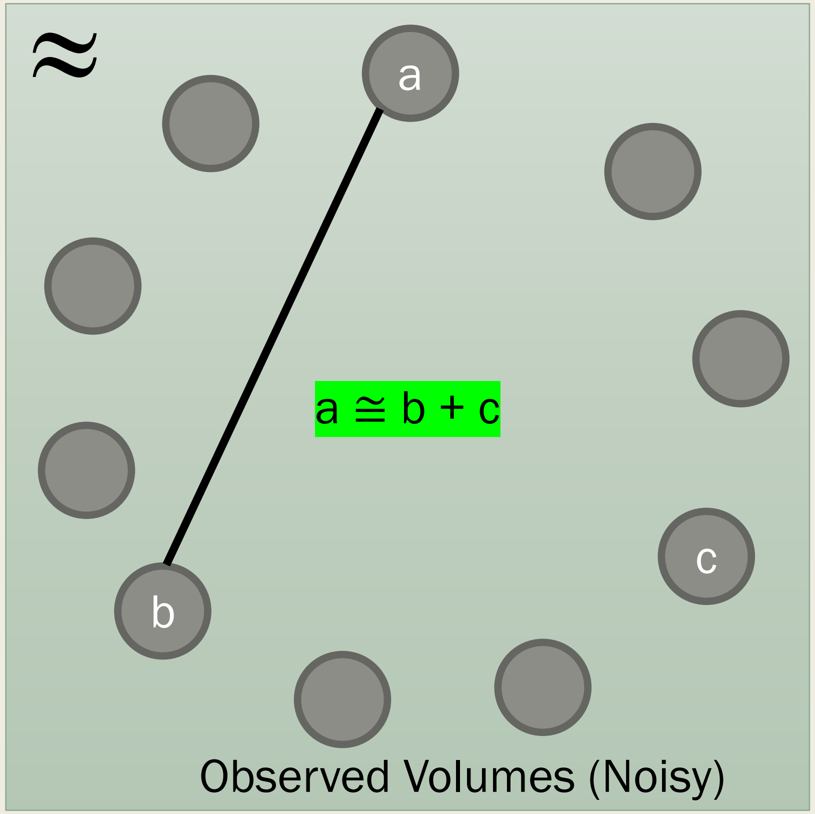
noise parameter

Recovered Volume : c

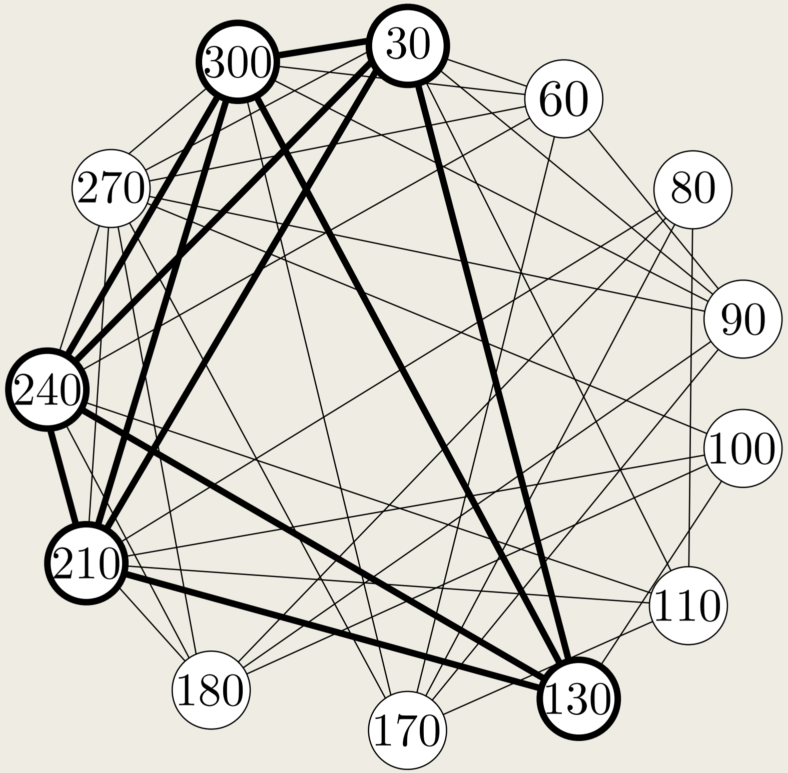
True Volume

w(c)

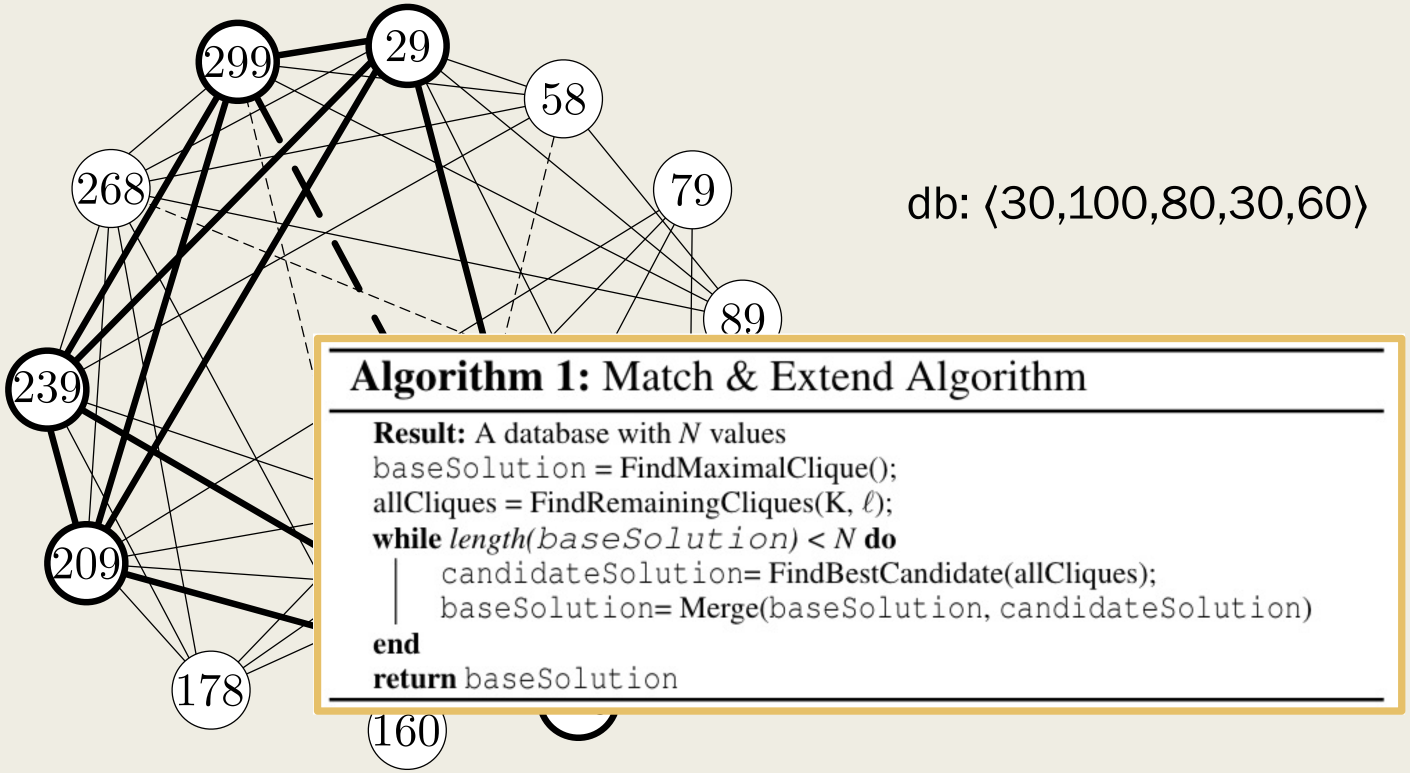
Offline Phase of the Attack



2. Extend the Clique Finding Algorithm



No Noise

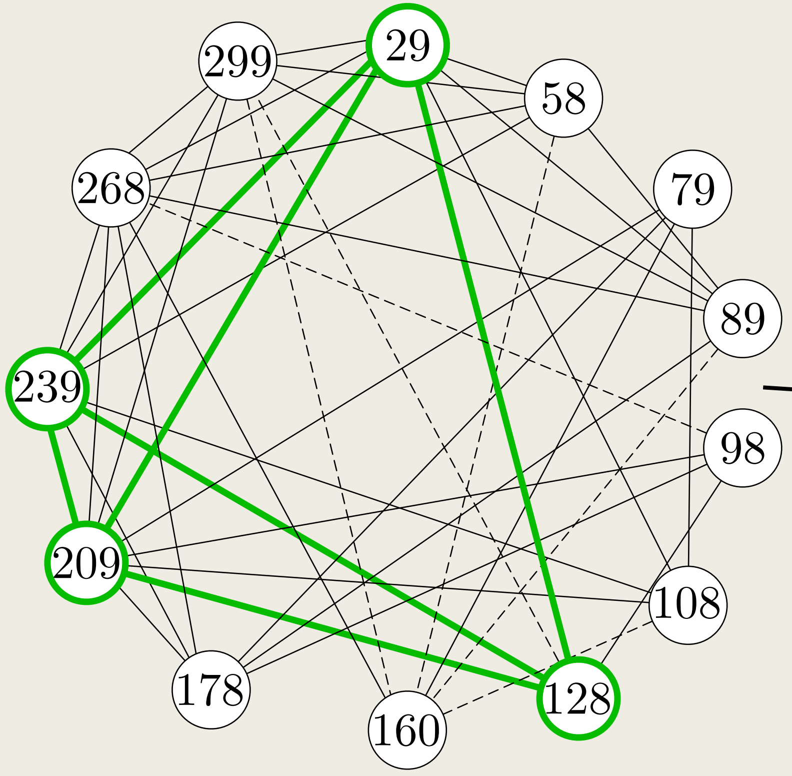


```
Algorithm 1: Match & Extend Algorithm
Result: A database with $N$ values
baseSolution = FindMaximalClique();
allCliques = FindRemainingCliques(K, $\ell$);
while length(baseSolution) < $N$ do
    candidateSolution= FindBestCandidate(allCliques);
    baseSolution= Merge(baseSolution, candidateSolution)
end
return baseSolution
```

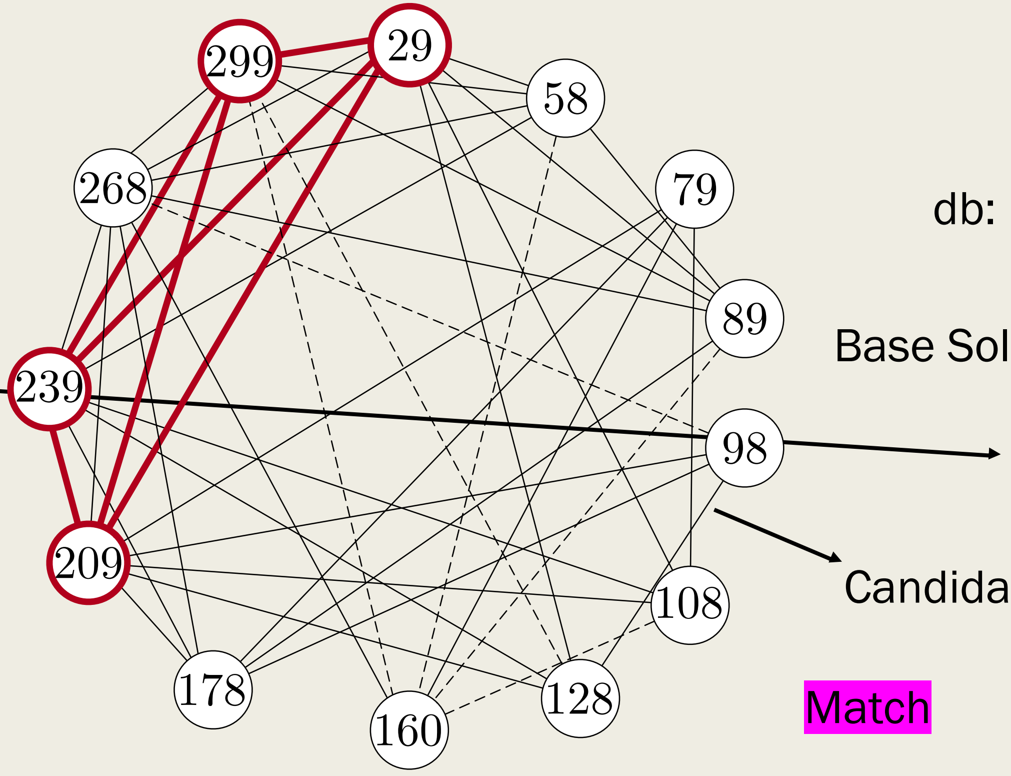
Noisy Measurements

db: ⟨30,100,80,30,60⟩

2. Extend the Clique Finding Algorithm*



Clique Found



Another Clique Found

db: ⟨30,100,80,30,60⟩

Base Solution:

⟨29,99,81,30⟩

Candidate Solution:

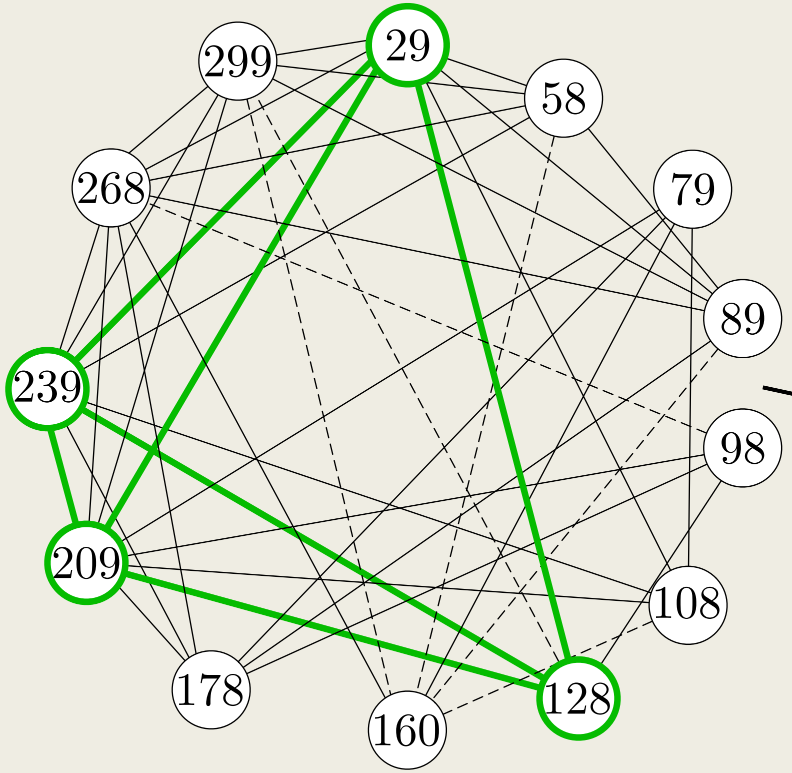
Match ⟨29,180,30,60⟩

Extend ⟨29, 99,81, 30,60⟩

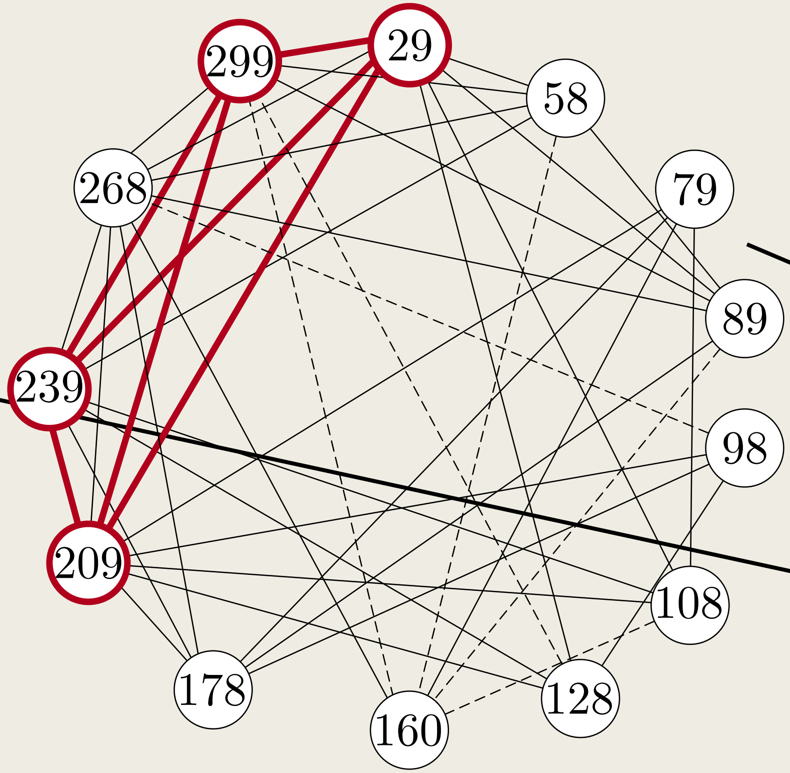
Approximate LCS

[*] Github repository: https://github.com/ariashahverdi/database_reconstruction

2. Extend the Clique Finding Algorithm*



Clique Found



Another Clique Found

db: ⟨30,100,80,30,60⟩

Base Solution:

⟨29,180,30,60⟩

Candidate Solution:

Match ⟨29,99,81,30⟩

Extend ⟨29, 99,81, 30,60⟩

Approximate LCS

[*] Github repository: https://github.com/ariashahverdi/database_reconstruction

Our Algorithmic Contribution

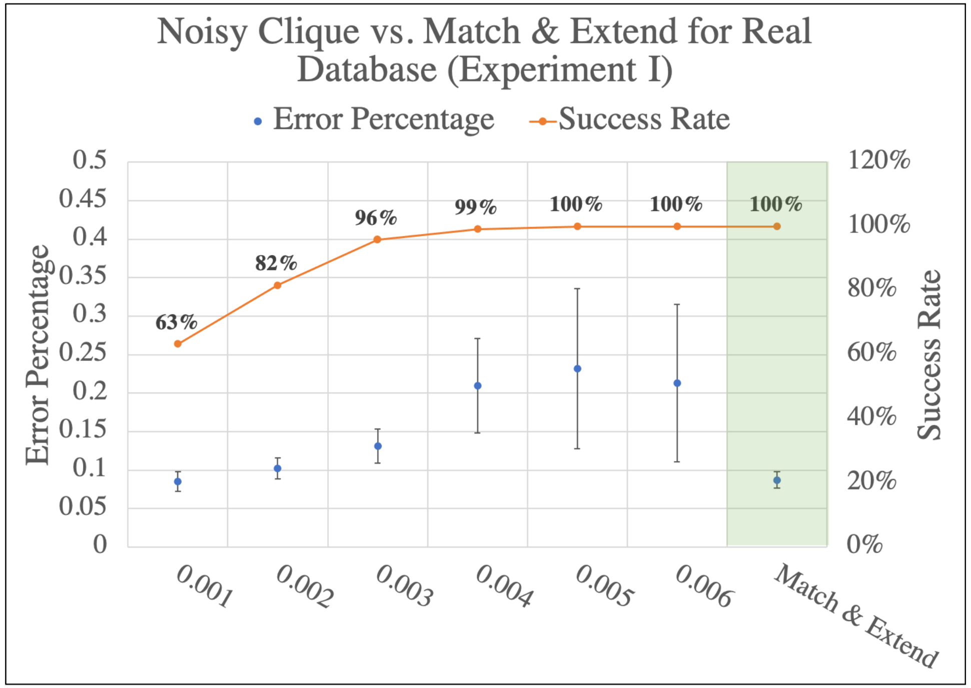
- ***Noisy Clique :*** Increase the Noise Budget
  - Pros
    - *More edges are connected in the graph*
  - Cons
    - *There might be some edges that connected by mistake (Especially if the size of the window gets too large)*
    - *The graph is getting bigger, hence the clique finding algorithm will takes longer time*
- ***Match & Extend:*** Fix the Noise Budget and combine multiple solutions

Experimental Setting

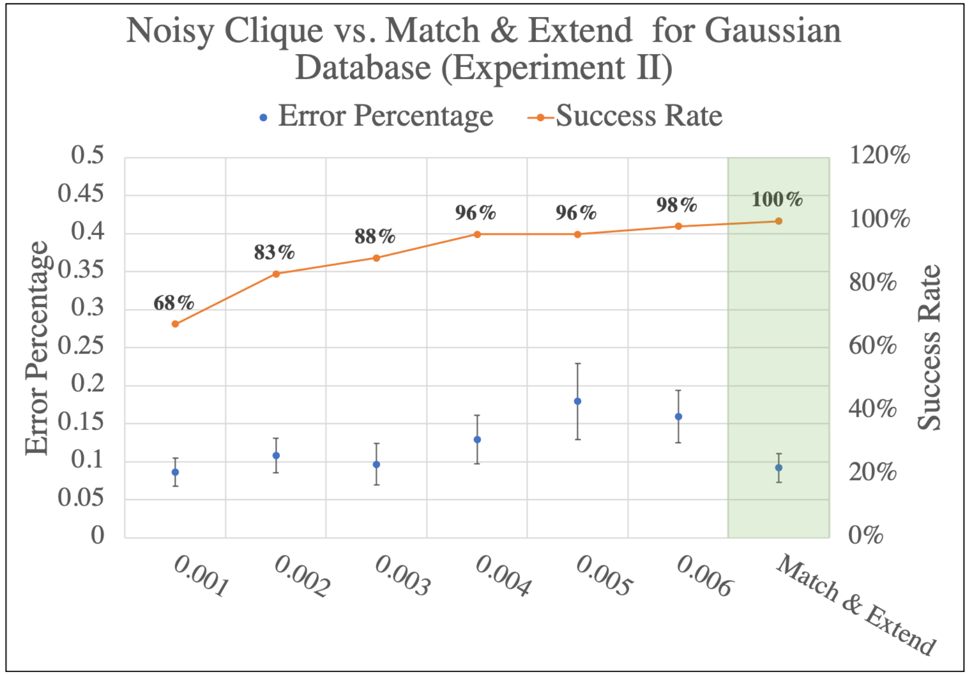
- Used Nationwide Inpatient Sample (NIS) from Healthcare Cost and Utilization Project (HCUP)
  - Randomly selected 100,000 records
- Performed range queries on the AMONTH (Jan-Dec) attribute

Experiments	*Query*	*Database*	*Notes*
I	Uniform	Real Database	
II	Uniform	Synthetic Database (Gaussian Like)	
III	Uniform	Real Database	Extra load present
IV	Non-Uniform	Real Database	
V	Uniform	Real Database	Some volumes are missing

Experimental Results

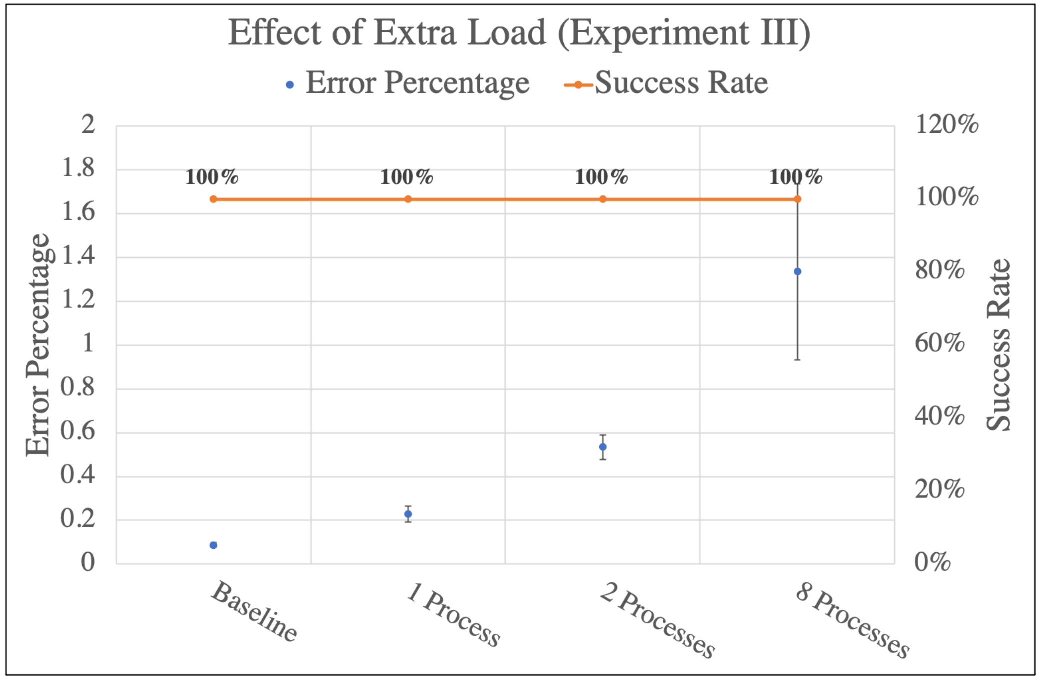


Real Database - Uniform Query

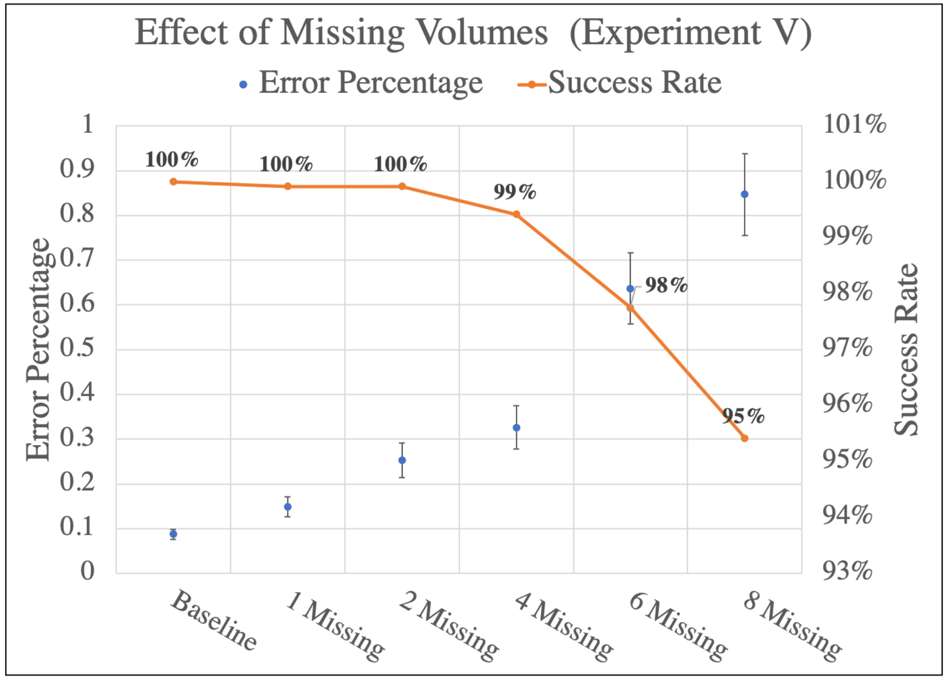


Synthetic Database - Uniform Query

Experimental Results



Real Database – Extra Load on The System



Real Database – Missing Volumes

Outline

- Overview of cache side-channel attacks
- Database Reconstruction from Noisy Volumes: A Cache Side-Channel Attack on SQLite.
  A. Shahverdi, M. Shirinov, D. Dachman-Soled.
  USENIX 2021
- How to Own the NAS in Your Spare Time.
  S. Hong, M. Davinroy, Y. Kaya, D. Dachman-Soled, T. Dumitras.
  ICLR 2020
  - Security analysis of deep neural networks operating in the presence of cache side-channel attacks.

  Sanghyun Hong, Michael Davinroy, Yiğitcan Kaya, Stuart Nevans Locke, Ian Rackow, Kevin Kulda, Dana Dachman-Soled, Tudor Dumitras, arXiv 2018.

Unique Architectures Are Costly To Obtain

- Neural architecture search (NAS) takes thousands of GPU hours
  - NASNet[1] search used **500 GPUs for 4 days** (CIFAR-10)
  - Prior work[2] used **800 GPUs for 28 days** (CIFAR-10)

[1] Zoph et al., *Learning Transferable Architectures for Scalable Image Recognition*, CVPR'17
[2] Zoph et al., *Neural Architecture Search with Reinforcement Learning*, ICLR'17

Unique Architectures Are Costly To Obtain

- Neural architecture search (NAS) takes thousands of GPU hours
  - NASNet[1] search used **500 GPUs for 4 days** (CIFAR-10)
  - Prior work[2] used **800 GPUs for 28 days** (CIFAR-10)
- Hand-crafting unique architectures require ML experts' effort
  - MalConv discussed **many failed architectures** in their paper
  - **10-15 ML experts** were required to design a new architecture for ImageNet

[1] Zoph et al., *Learning Transferable Architectures for Scalable Image Recognition*, CVPR'17

[2] Zoph et al., *Neural Architecture Search with Reinforcement Learning*, ICLR'17

Unique Architectures Are Costly To Obtain

- Neural architecture search (NAS) takes thousands of GPU hours
  - NASNet[1] search used **500 GPUs for 4 days** (CIFAR-10)
  - Prior work[2] used **800 GPUs for 28 days** (CIFAR-10)
- Hand-crafting unique architectures require ML experts' effort
  - MalConv discussed **many failed architectures** in their paper
  - **10-15 ML experts** were required to design a new architecture for ImageNet

They Become Intellectual Property or Trade Secrets

[1] Zoph et al., *Learning Transferable Architectures for Scalable Image Recognition*, CVPR'17
[2] Zoph et al., *Neural Architecture Search with Reinforcement Learning*, ICLR'17

What If Your Unique DL Architectures Is Stolen?

What Benefit Can An Adversary Have?

- Using the *stolen architecture*:
  - The attacker can **train a functional model** that has the same accuracy

[1] So et al., *Evolved Transformer,* ICML19

What Benefit Can An Adversary Have?

- Using the *stolen architecture*:
  - The attacker can **train a functional model** that has the same accuracy
  - The attacker can train a high-performing model **even on a different dataset**[1]

[1] So et al., *Evolved Transformer,* ICML19

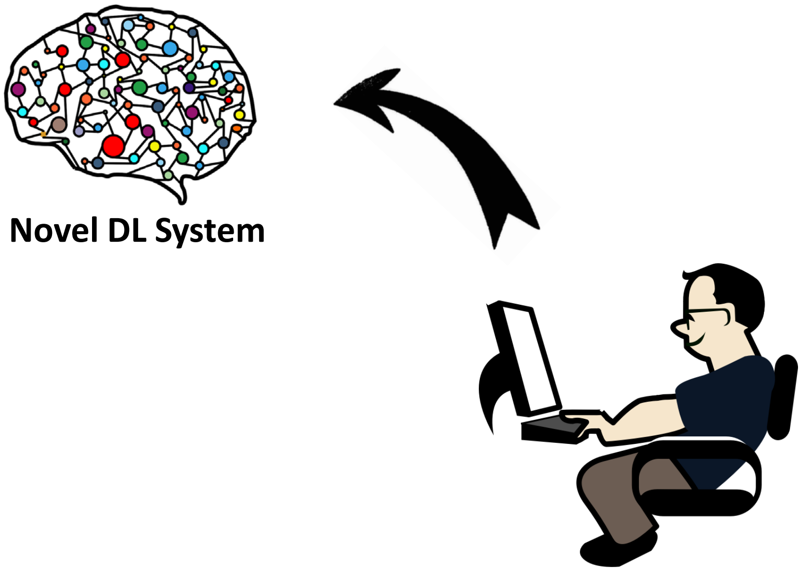
What Benefit Can An Adversary Have?

- Using the *stolen architecture*:
  - The attacker can **train a functional model** that has the same accuracy
  - The attacker can train a high-performing model **even on a different dataset**[1]
  - The adversary can **perform further attacks**[2] exploiting data augmentation

[1] So et al., *Evolved Transformer,* ICML19

[2] Xiao et al., *Seeing Is Not Believing: Camouflage Attacks on Image Scaling Algorithm,* USENIX'19

What Is Our Threat Model?

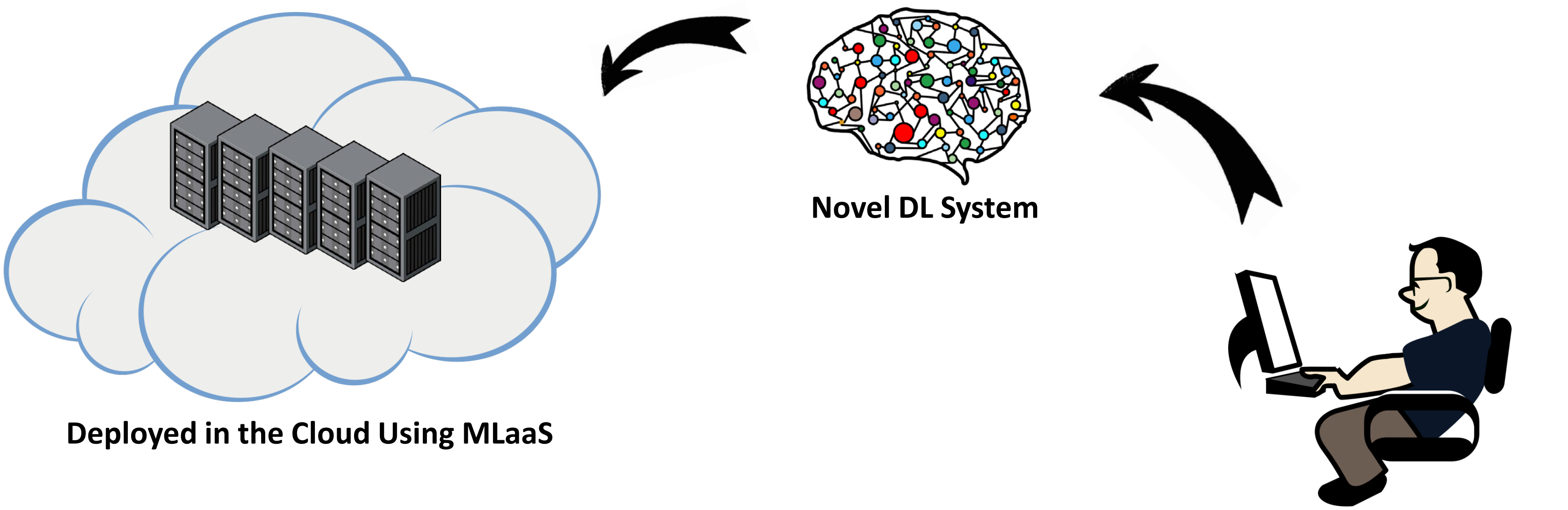


Novel DL System

Researchers and practitioner

What Is Our Threat Model?

- Machine-Learning-as-a-Service (MLaaS)



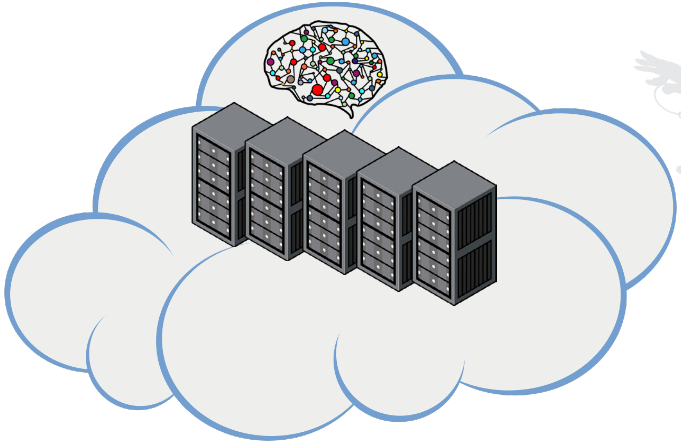
Novel DL System

Deployed in the Cloud Using MLaaS

Researchers and practitioner

What Is Our Threat Model?

- **In MLaaS**: Physical access[1] to the hardware is impractical



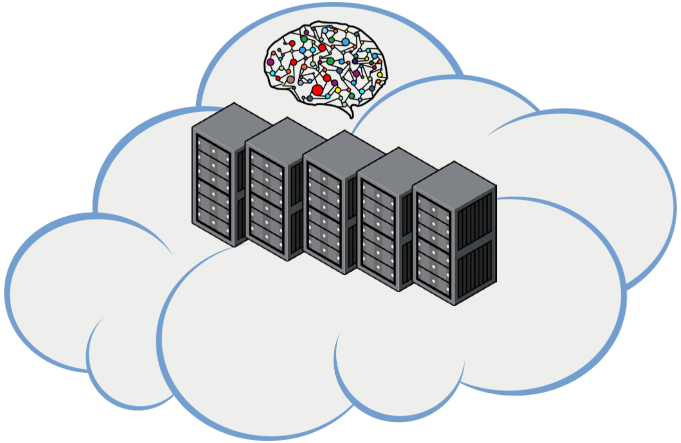
Deployed in the Cloud Using MLaaS

Researchers and practitioners

[1] Batina et al., *CSI NN: Reverse Engineering of Neural Network Architectures through Electromagnetic Side Channel,* USENIX'19

What Is Our Threat Model?

- **In MLaaS**: remote hardware side-channel attacks make this practical



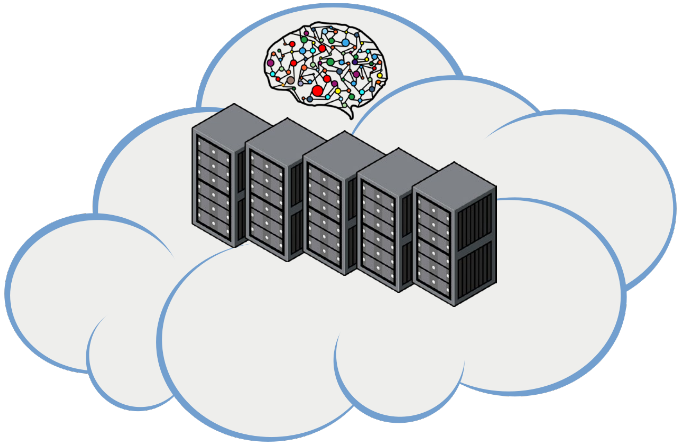
Deployed in the Cloud Using MLaaS

Remote Side-Channel Attacker

Researchers and practitioners

What Is Our Threat Model?

- **In MLaaS**: remote hardware side-channel attacks make this practical



Deployed in the Cloud Using MLaaS

w/o Direct Queries

in Contrast to Prior Work[2]

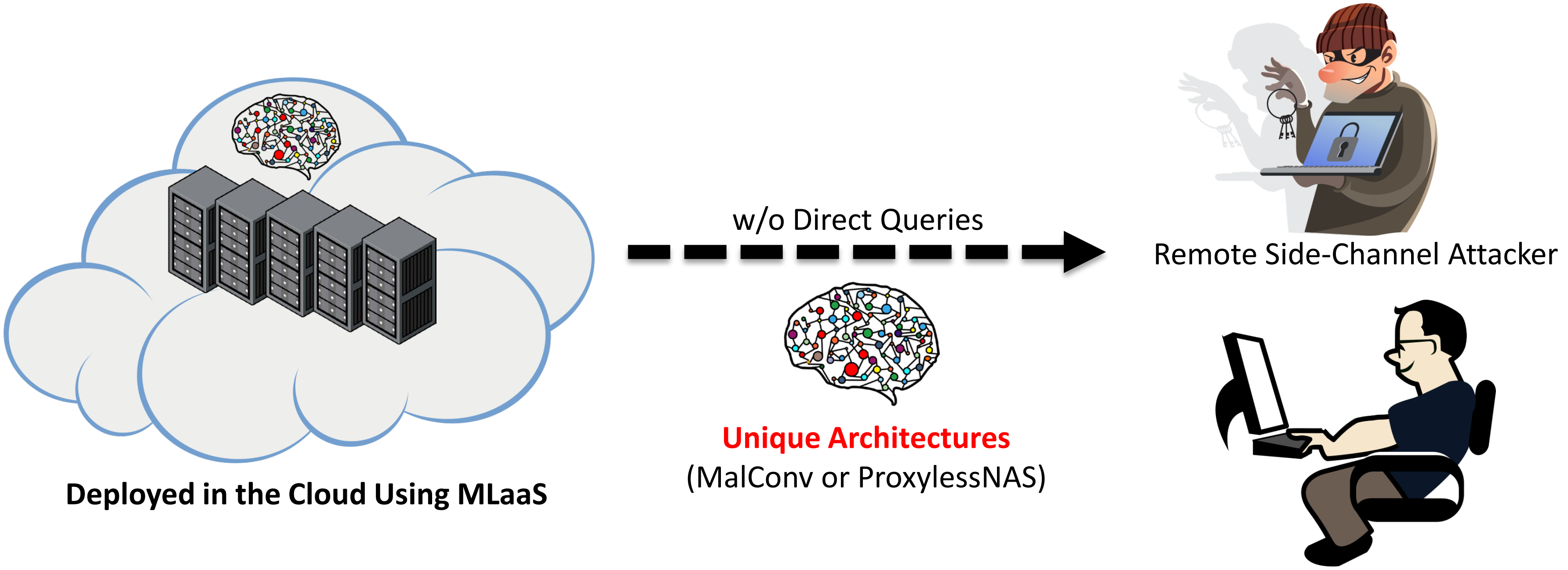
Remote Side-Channel Attacker

Researchers and practitioners

[2] Tramer et al., *Stealing Machine Learning Models via Prediction APIs, USENIX'16*

What Is Our Threat Model?

- Our attack steals the **unique architectures**



w/o Direct Queries

Remote Side-Channel Attacker

Deployed in the Cloud Using MLaaS

Unique Architectures
(MalConv or ProxylessNAS)

Researchers and practitioners

Our Reconstruction Attack

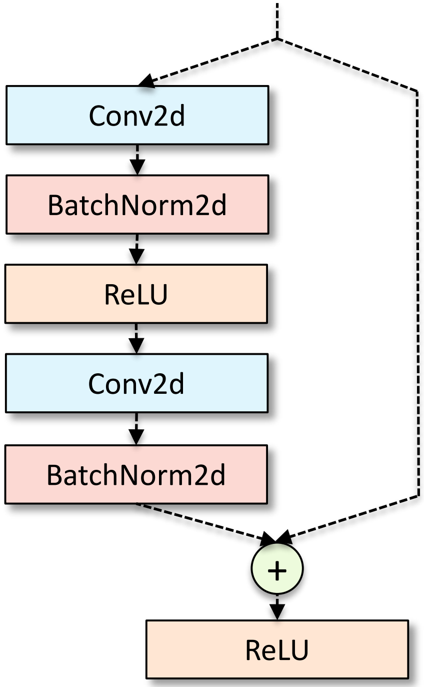
1. Identify the DL computations to monitor
2. Monitor the DL computations via Flush+Reload
3. De-noise the Flush+Reload trace
4. Profile the computation times
5. Perform the reconstruction process

Our Reconstruction Attack

1. Identify the DL computations to monitor
2. Monitor the DL computations via Flush+Reload
3. De-noise the Flush+Reload trace
4. Profile the computation times
5. Perform the reconstruction process

How Does the Flush+Reload Trace Look Like?

A Residual Block for ResNets

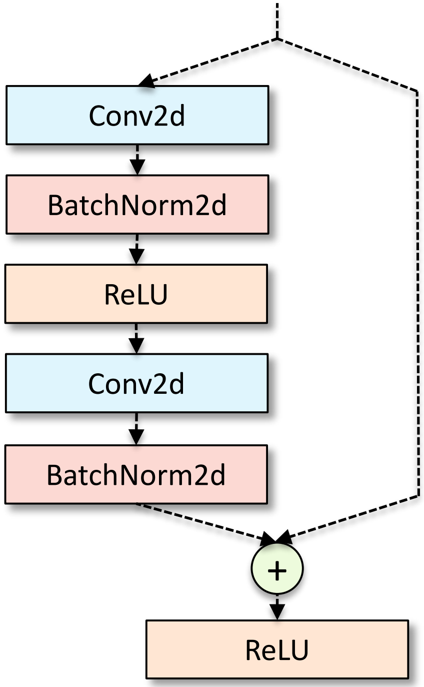


Flush+Reload Trace

[1] Conv2d,	$t_1, 1, n_1$
[2] BatchNorm2d,	$t_2, 1, n_2$
[3] ReLU,	$t_3, 1, n_3$
[4] Conv2d,	$t_4, 1, n_4$
[5] BatchNorm2d,	$t_5, 1, n_5$
[6] add,	$t_6, 1, n_6$
[7] ReLU,	$t_7, 1, n_7$

Reconstruction Attacks in Prior Work

A Residual Block for ResNets



Flush+Reload Trace

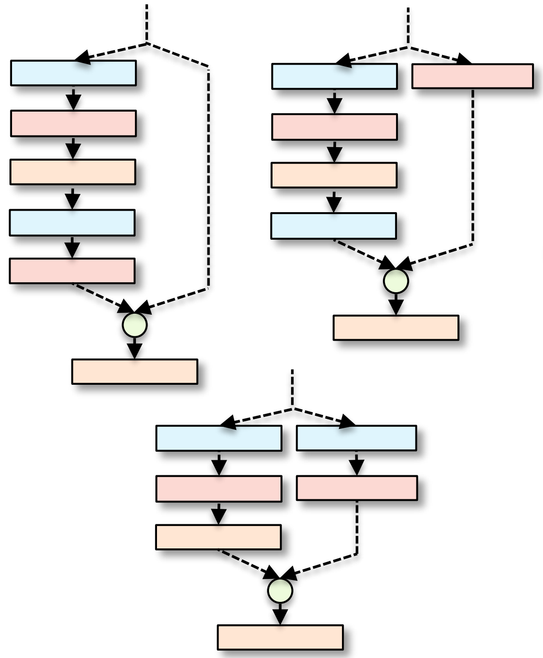
[1] Conv2d,	$t_1, 1, n_1$
[2] BatchNorm2d,	$t_2, 1, n_2$
[3] ReLU,	$t_3, 1, n_3$
[4] Conv2d,	$t_4, 1, n_4$
[5] BatchNorm2d,	$t_5, 1, n_5$
[6] add,	$t_6, 1, n_6$
[7] ReLU,	$t_7, 1, n_7$

Prior work[1] assumes the attacker knows it's ResNet - Easy

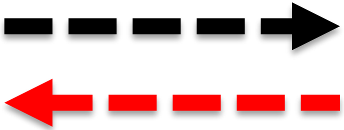
[1] Hong et al., *Security Analysis of Deep Neural Networks Operating in the Presence of Cache Side-Channel Attacks*, arXiv'18

What If The Attacker Doesn't Know It's ResNet?

???



Flush+Reload Trace

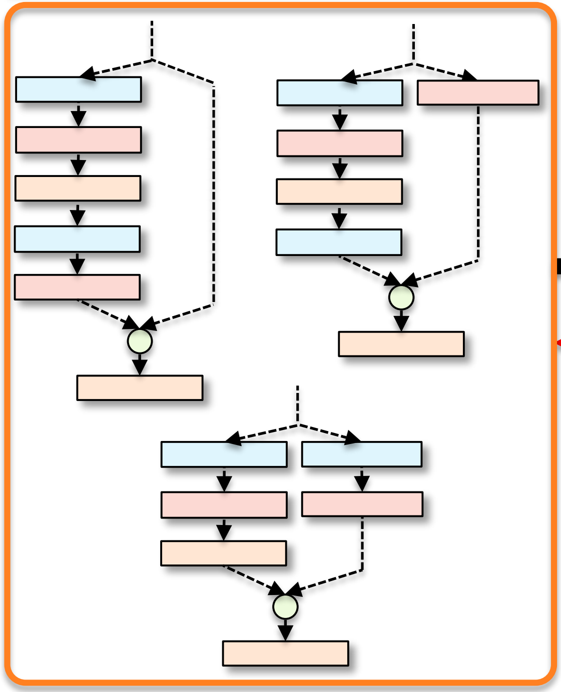


[1] Conv2d, $t_1, 1, n_1$
[2] BatchNorm2d, $t_2, 1, n_2$
[3] ReLU, $t_3, 1, n_3$
[4] Conv2d, $t_4, 1, n_4$
[5] BatchNorm2d, $t_5, 1, n_5$
[6] add, $t_6, 1, n_6$
[7] ReLU, $t_7, 1, n_7$

Problem: There are **multiple interpretations** of the trace

Our Reconstruction Attack – Generation

???



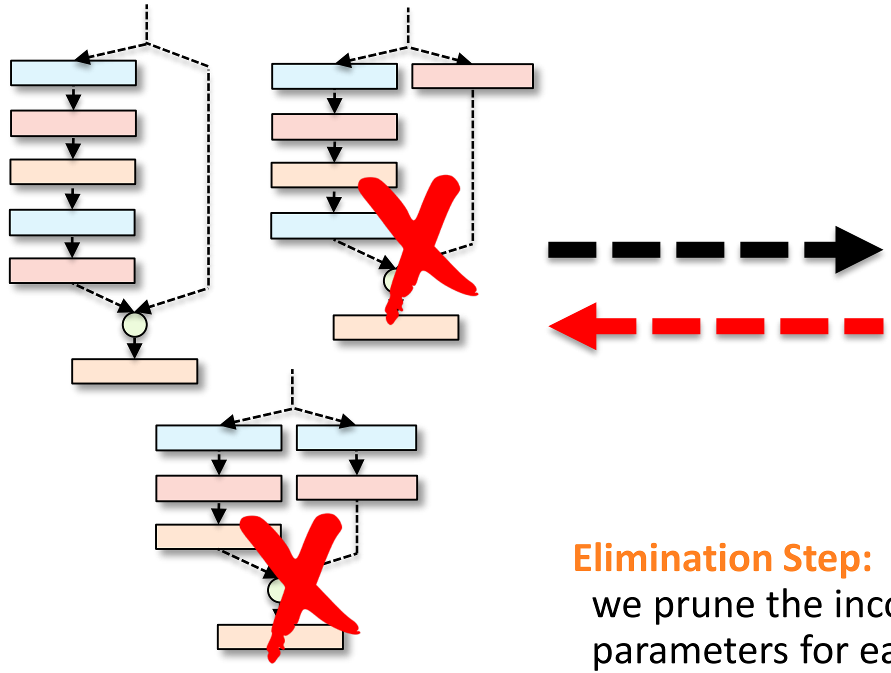
Flush+Reload Trace

[1] Conv2d,	$t_1, 1, n_1$
[2] BatchNorm2d,	$t_2, 1, n_2$
[3] ReLU,	$t_3, 1, n_3$
[4] Conv2d,	$t_4, 1, n_4$
[5] BatchNorm2d,	$t_5, 1, n_5$
[6] add,	$t_6, 1, n_6$
[7] ReLU,	$t_7, 1, n_7$

Generation Step: we create all the possible candidate architectures

Our Reconstruction Attack – Elimination

???



Flush+Reload Trace

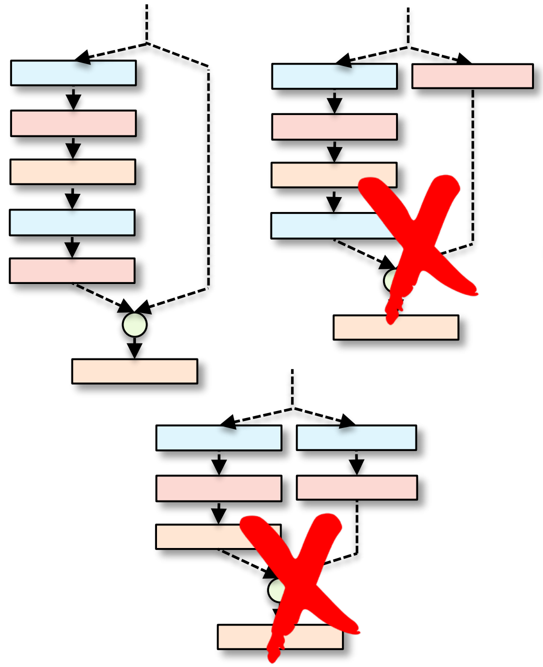
[1] Conv2d, $t_1, 1, n_1$
[2] BatchNorm2d, $t_2, 1, n_2$
[3] ReLU, $t_3, 1, n_3$
[4] Conv2d, $t_4, 1, n_4$
[5] BatchNorm2d, $t_5, 1, n_5$
[6] add, $t_6, 1, n_6$
[7] ReLU, $t_7, 1, n_7$

Elimination Step:

we prune the incompatible candidates by estimating computation parameters for each layer based on the timing information

Our Reconstruction Attack – Elimination

???



Flush+Reload Trace

[1] Conv2d, t_1, 1, n_1
[2] BatchNorm2d, t_2, 1, n_2
[3] ReLU, t_3, 1, n_3
[4] Conv2d, t_4, 1, n_4
[5] BatchNorm2d, t_5, 1, n_5
[6] add, t_6, 1, n_6
[7] ReLU, t_7, 1, n_7

Computation time $(t_i - t_{i-1})$
$\propto$ the size of matrix multiplications

Elimination Step:

we prune the incompatible candidates by estimating computation parameters for each layer based on the timing information

Evaluation Result

	MalConv	ProxylessNAS-CPU
# candidates	20	180,244
# compatible architectures	1	1
Reconstruction error	**0 %**	**0 %**
Time taken	< 10 CPU minutes	< 12 CPU hours

Our attack accurately reconstructs unique architectures

Conclusion and Future Work

- **Conclusion:** Our attack can reconstruct unique architectures precisely Unique architectures can be stolen by our reconstruction attack

- **Future Work:** Countermeasures against the reconstruction attacks

MARYLAND
CYBERSECURITY CENTER