

# *From Shared-All to Shared-Nothing*

Successfully used Patterns in  
application and table design  
with Hbase

Bob Schulze, eCircle AG

March 2010 @ Berlin Apache Hadoop Get Together

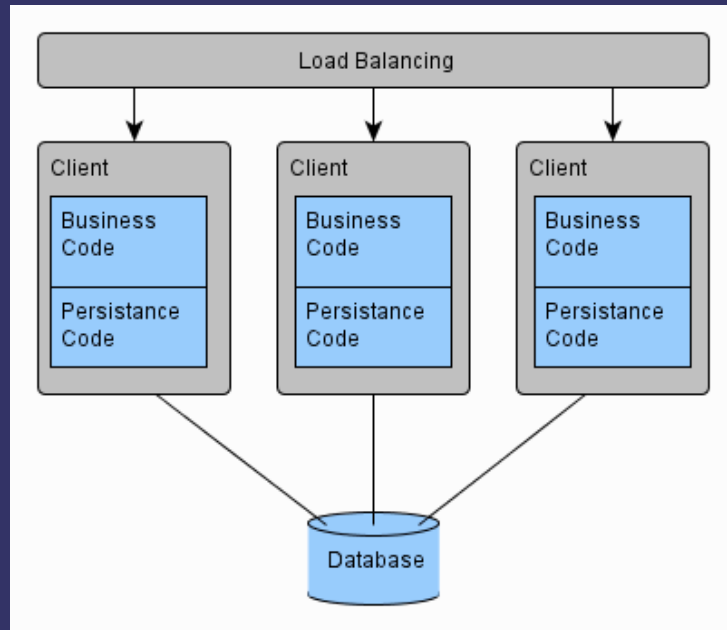
# *Audience*

- ⇒ You have Big Data
- ⇒ Your Organization needs predictable scaling options
- ⇒ You need to be flexible with your Data
- ⇒ You are a Techie Person

# Content

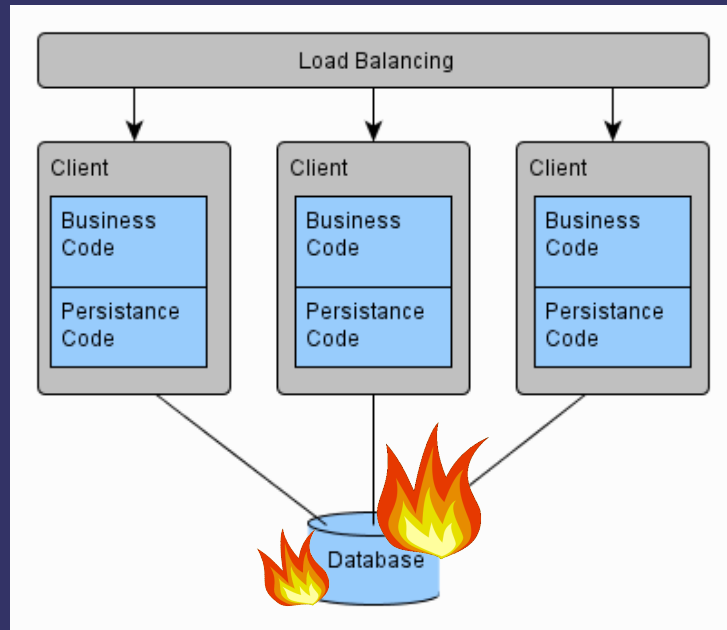
- ⇒ What is shared?
- ⇒ Recap RDBMS vs HBase/BigTable
- ⇒ Example: Credit-Card Processing
- ⇒ Storage Patterns
- ⇒ Application Design Proposal
- ⇒ Missing: Call for Features
- ⇒ HbaseExplorer

# Shared ..somewhat



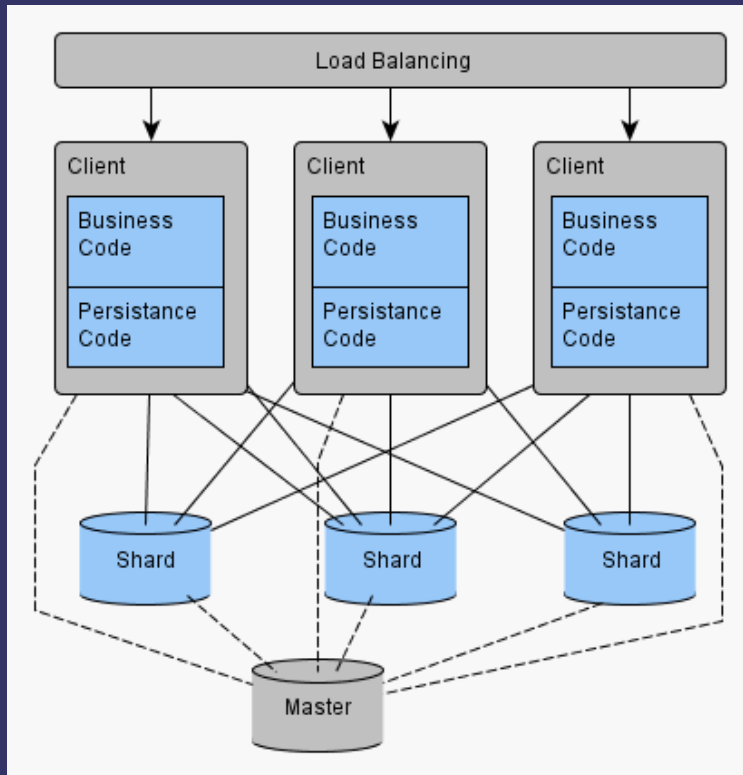
- ➔ Representation and Business Layers have well known scaling patterns
- ➔ Many of these patterns rely on a transactional database underneath
- ➔ Distributed transactions are expensive

# Shared ..somewhat



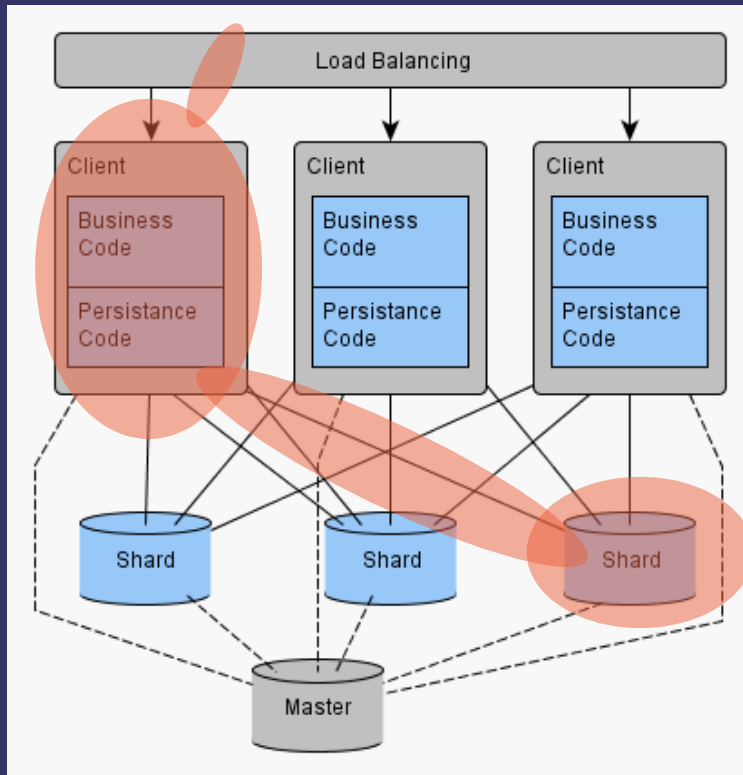
- ➔ Representation and Business Layers have well known scaling patterns
- ➔ Many of these patterns rely on a transactional database underneath
- ➔ Distributed transactions are expensive

# Shared ..almost nothing



- ➔ All Business Threads can run independently a long way
- ➔ Contention only within one Shard → More Shards, Less Contention
- ➔ Highly Available and Consistent
- ➔ Give away perfection: Transactions must be hand-made

# Shared ..almost nothing



- ➔ All Business Threads can run independently a long way
- ➔ Contention only within one Shard → More Shards, Less Contention
- ➔ Highly Available and Consistent
- ➔ Give away perfection: Transactions must be hand-made

# Content

- ⇒ What is shared?
- ⇒ Recap RDBMS vs HBase/BigTable
- ⇒ Example: Credit-Card Processing
- ⇒ Storage Patterns
- ⇒ Application Design Proposal
- ⇒ Missing: Call for Features
- ⇒ HbaseExplorer



# RDBMs Solutions

Card	Dealer	Amount	Location	Currency
Customer	Card	Name	Address	Email
Card	Model	Valid From		
Customer	Date	Supporter	Req-ID	Status

- ➔ Transactions
- ➔ Access Rules
- ➔ Types
- ➔ FK's
- ➔ Fixed structure
- ➔ History?
- ➔ Rows?

# *Alternative: HBase*

- ⇒ Key/Value with structure in the „fat“ values
  - Arbitrary keys, but retain sortability
- ⇒ Data stored in Shards (regionserver) by splitting up the key range
- ⇒ Values are organized in Families
- ⇒ All Data has history

# Hbase: Key-Value

Card Number  
1233.45.33-23

TS	Card	Transaction	Owner	Support	Notes
t5	registerCode=<..> model=<model> pinHash=<md5(pin)> ValidFrom=<md5(f)> ValidTo=<...>				A new card was issued to the customer
t4		<DealerId>= <amount> Location=<..>	pinAttempts=<..>		A Transaction was made
t3				Reason=<...> <ReqId>= <status> Supporter=<..>	Support Request from Customer
t2			City=<...> Addressid=<..> src=byTel operator=<...>	Reason=<...> <ReqId>= Solved Supporter=<>	Address-Change by support
t1			Email= <md5(email)> src=web		Email-Change from WebSite

Card Number  
1233.45.33-24

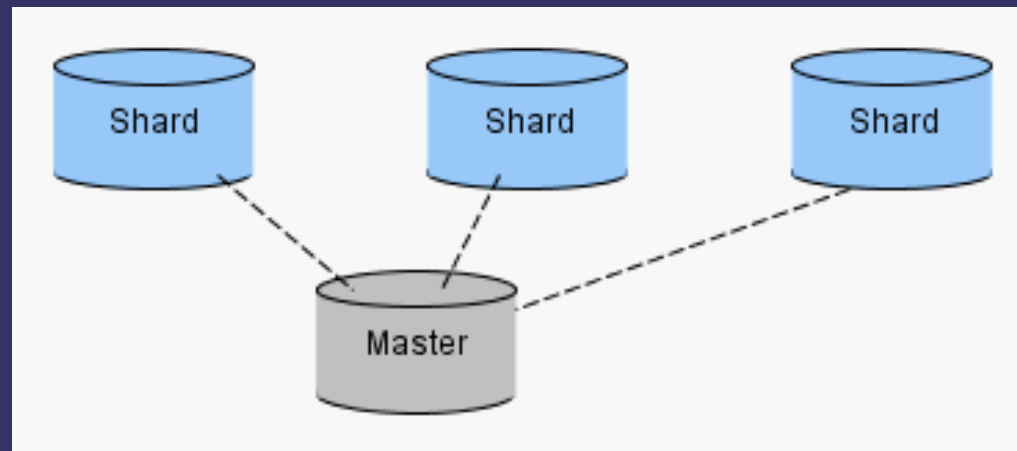
TS	Card	Transaction	Owner	Support	Notes
t5	registerCode=<..> model=<model> pinHash=<md5(pin)> ValidFrom=<md5(f)> ValidTo=<...>				A new card was issued to the customer
t4		<DealerId>= <amount> Location=<..>	pinAttempts=<..>		A Transaction was made
t3				Reason=<...> <ReqId>= <status> Supporter=<..>	Support Request from Customer

# *Hbase: Regionserver hold Shards*

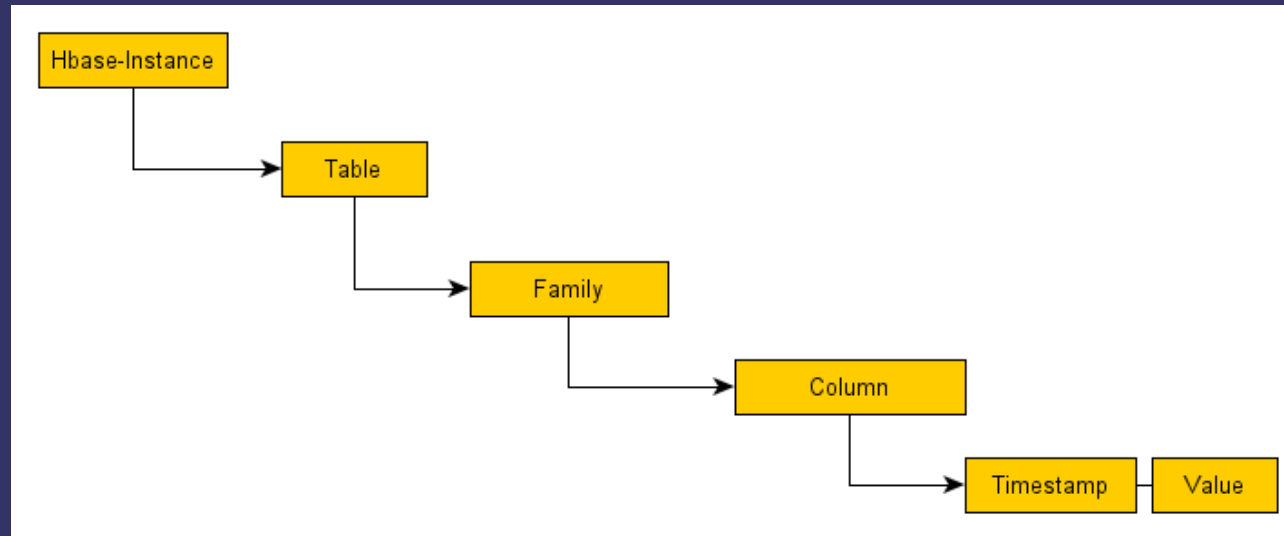
a...k

l...m

n...z



# Recap: Hbase/BigTable Data Model



⇒ family+Column=Column Qualifier

# Column Families

## ⇒ Stored in own Files

- Important for retrieval

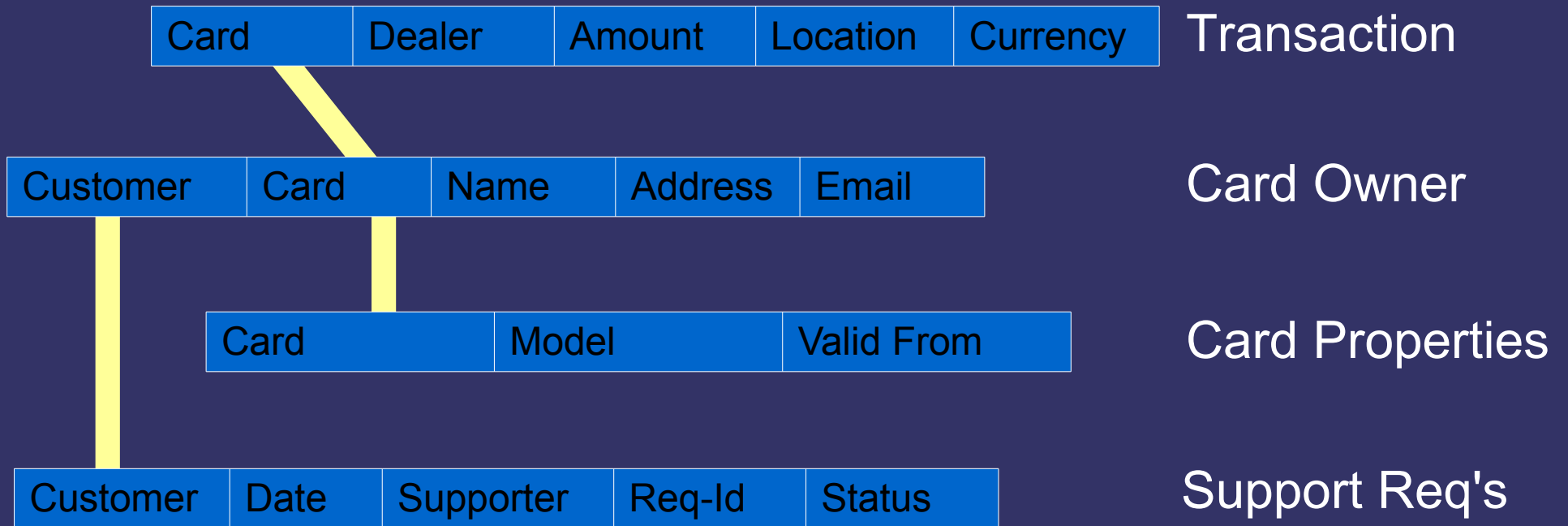
## ⇒ Have own Settings

- Compression
- Versions
- Timed Deletions (TTL)
- Size Constraints
- Counters

# Content

- ⇒ What is shared?
- ⇒ Recap RDBMS vs HBase/BigTable
- ⇒ Example: Credit-Card Processing
- ⇒ Storage Patterns
- ⇒ Application Design Proposal
- ⇒ Missing: Call for Features
- ⇒ HbaseExplorer

# Example: Credit Card Processing





# Hbase main table Layout

RowKey: Card Number

TS	Card	Transaction	Owner	Support	Notes
t5	registerCode=<..> model=<model> pinHash=<md5(pin)> ValidFrom=<md5(f)> ValidTo=<...>				<i>A new card was issued to the customer</i>
t4		<DealerId>= <amount> Location=<..>	pinAttempts=<..>		<i>A Transaction was made</i>
t3				Reason=<...> <ReqId>= <status> Supporter=<..>	<i>Support Request from Customer</i>
t2			City=<...> Addressid=<..> src=byTel operator=<...>	Reason=<...> <ReqId>= Solved Supporter=<>	<i>Address-Change by support</i>
t1			Email= <md5(email)> src=web		<i>Email-Change from WebSite</i>

# Supplementary (index-) Tables

## ➔ Email to Card Mapping

RowKey: md5(email)

Card
<card>
<card>

Allows to do lookups by a given email  
Index can be maintained without transaction!

## ➔ Address references

RowKey: addressid

Address
Name=<name>
City=<city>
Street=<street>

Sample for a simple relation  
Can be further encrypted

# Content

- ⇒ What is shared?
- ⇒ Recap RDBMS vs HBase/BigTable
- ⇒ Example: Credit-Card Processing
- ⇒ **Storage Patterns**
- ⇒ Application Design Proposal
- ⇒ Missing: Call for Features
- ⇒ HbaseExplorer

# *Patterns to Store Data, why?*

- ⇒ We can talk about it
- ⇒ Based on Space efficiency
  - Even if space is cheap, data has to be searched through and has to be moved
- ⇒ Based on Lookup efficiency
  - Most Data is stored sorted
- ⇒ Used for direct lookups as well as in MR aggregations

# Pattern: swim-above

➔ Most recent value at top (in API)

- Where was the last Transaction?

get(Cardid: 123.22.34-24, Family: Transaction, Column: Location)

TS	Card	Transaction	Owner	Support	Notes
t5	registerCode=123 model=superFlash pinHash=aw3224hhds ValidFrom=se344qq1 ValidTo=12esdrf43q.q				<i>A new card was issued to the customer</i>
t4		D123376=123E Location=Berlin	pinAttempts=1		<i>A Transaction was made</i>
t3		D2231=82.22E Location=Munich	pinAttempts=2		<i>A Transaction was made</i>

- What is the current model?

get(Cardid: 123.22.34-24, Family: Card, Column: model)

# Pattern: swim-above

➔ Most recent value at top (in API)

- Where was the last Transaction?

get(Cardid: 123.22.34-24, Family: Transaction, Column: Location)

TS	Card	Transaction	Owner	Support	Notes
t5	registerCode=123 model=superFlash pinHash=aw3224hhds ValidFrom=ss011qq1 ValidTo=12esdrf43q.q				A new card was issued to the customer
t4		D123370-123E Location=Berlin	pinAttempts=1		A Transaction was made
t3		D2231=82.22E Location=Munich	pinAttempts=2		A Transaction was made

- What is the current model?

get(Cardid: 123.22.34-24, Family: Card, Column: model)

# Pattern: Data grouped by Timestamp

- Who changed the Address to „Munich“ ?

1. Figure out the Timestamp(s)

`get(Cardid: 123.22.34.24, City=Munich) → ts2`

2. get the fields for this TS

`get(Cardid: 123.22.34-24, timestamp: t2)`

TS	Card	Transaction	Owner	Support	Notes
t3				Reason=<...> <Reqld>= <status> Supporter=<..>	<i>Support Request from Customer</i>
t2			City=Munich Addressid=<..> src=byTel operator=<...>	Reason=Call-In R213=Solved Supporter=PaulG	<i>Address-Change by support</i>
t1			Email= <md5(email)> src=web		<i>Email-Change from WebSite</i>

# Pattern: Data grouped by Timestamp

- Who changed the Address to „Munich“ ?

1. Figure out the Timestamp(s)

get(Cardid: 123.22.34.24, City=Munich) → ts2

2. get the fields for this TS

get(Cardid: 123.22.34-24, timestamp: t2)

TS	Card	Transaction	Owner	Support	Notes
t3				Reason=<...> <Reqld>= <status> Supporter=<..>	<i>Support Request from Customer</i>
t2			City=Munich Addressid=<..> src=byTel operator=<...>	Reason=Call-In R213=Solved Supporter=PaulG	<i>Address-Change by support</i>
t1			Email= <md5(email)> src=web		<i>Email-Change from WebSite</i>



# Pattern: ColumnName-Is-Value

➔ No value, Often useful for indexes

RowKey: md5(email)

Card

123.23662-21  
123.23452-24

RowKey: 123.23452-24

TS	Card	Transaction	Owner	Support	Notes
t3				Reason=<...> <ReqId>= <status> Supporter=<..>	<i>Support Request from Customer</i>
t2			City=Munich Addressid=<..> src=byTel operator=<...>	Reason=Call-In R213=Solved Supporter=PaulG	<i>Address-Change by support</i>

# Pattern: ColumnName-Is-Value

⇒ No value, Often useful for indexes

RowKey: md5(email)

Card

123.23662-21

123.23452-24

RowKey: 123.23452-24

TS	Card	Transaction	Owner	Support	Notes
t3				Reason=<...> <Reqld>= <status> Supporter=<..>	<i>Support Request from Customer</i>
t2			City=Munich Addressid=<..> src=byTel operator=<...>	Reason=Call-In R213=Solved Supporter=PaulG	<i>Address-Change by support</i>

- Where did al.lias@gmx.com use his cards?

index.get(9fc81d4292e6a404c2d64c9eaa66e43a) → Cardids  
cards.get(123.23452-24,...)

# Pattern: Column-Enum

- What is the status of Support-Request R213

(status is one of: Opened, Reviewed, Assigned, Pending, Solved)

get(Cardid: 123.22.34.24, Family: Support, Column: R213)

TS	Card	Transaction	Owner	Support	Notes
t3				Reason=<...> <Reqld>= <status> Supporter=<..>	<i>Support Request from Customer</i>
t2			City=Munich Addressid=<..> src=byTel operator=<...>	Reason=Call-In R213=Solved Supporter=PaulG	<i>Address-Change by support</i>
t1			Email= <md5(email)> src=web		<i>Email-Change from WebSite</i>

# *Pattern: Atomic Counters*

- Solves the common Problem when many clients try to update one/few rows in a RDBMS table
- Use a separate table/family/column, use the key or family to partition the load
- Example: Write a Record and add up some stats
  1. `cards.insert(key: 123.22.34.24, Column: R213, Value=Solved,...)`
  2. `stats.increment(key: 123.22.34.24, Column: SREGSPERMONTH,+1)`
- Small overhead even on excessive use:
  - `<terminal-id>:year-month-date=<cnt>`
  - `<terminal-id>:year-month=<cnt>`
- timestamps!

# Pattern: index table

## ⇒ Constant Costs

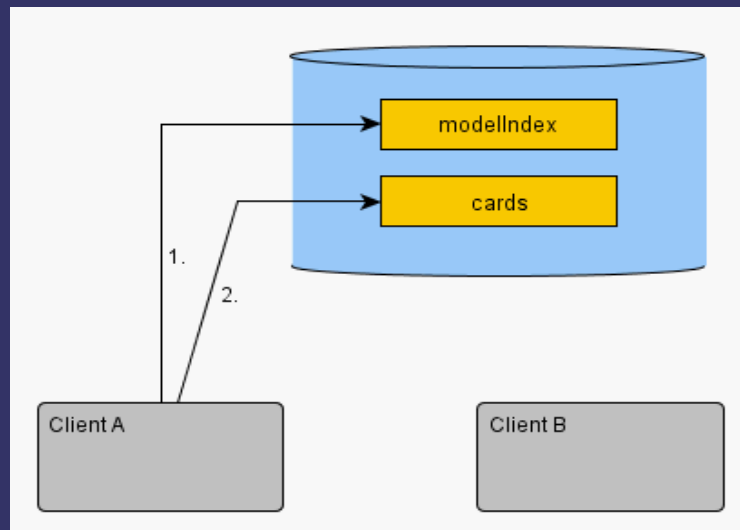
- Always one more insert to (another sharded) index table

## ⇒ Lock Free

- Versions=1

## ⇒ Only „eventually“ consistent

- But on our control!



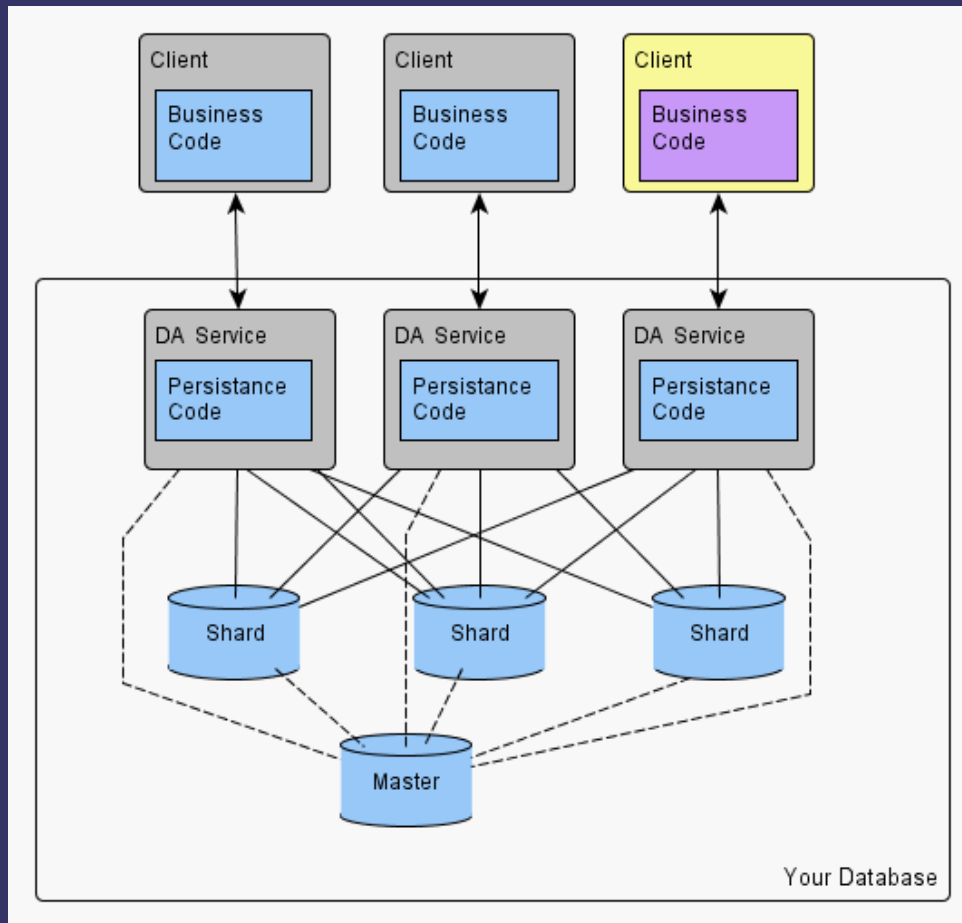
# *Pattern Summary*

- ⇒ Swim-above
- ⇒ Data grouped by timestamp
- ⇒ ColumnName-is-Value
- ⇒ Column-Enum
- ⇒ Atomic Counter
- ⇒ Index table

# Content

- ⇒ What is shared?
- ⇒ Recap RDBMS vs HBase/BigTable
- ⇒ Example: Credit-Card Processing
- ⇒ Storage Patterns
- ⇒ **Application Design Proposal**
- ⇒ Missing: Call for Features
- ⇒ HbaseExplorer

# Application Design



➔ Persistence Code moves out of App

- Gets reusable!
- Easy to test

➔ Fix what's missing

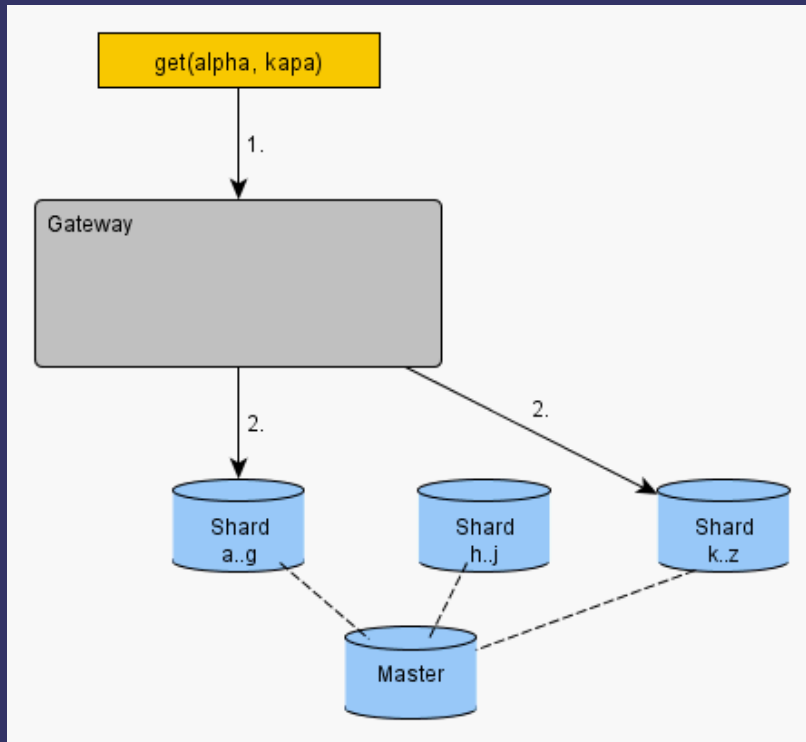
- Security / Access Control
- Firewall
- Index Handling
- Transactions
- ORM mappings



# Content

- ⇒ What is shared?
- ⇒ Recap RDBMS vs HBase/BigTable
- ⇒ Example: Credit-Card Processing
- ⇒ Storage Patterns
- ⇒ Application Design Proposal
- ⇒ **Missing: Call for Features**
- ⇒ HbaseExplorer

# Hbase: missing pieces



- ➔ Multi-Get/Scan would allow to read data from multiple Region Servers in parallel to one client
- ➔ Same with Multi-Put
- ➔ Patches avail., 0.21

# *Hbase: missing pieces*

- ⇒ Server Side Processing reduces data transfer and distributes computing
  - Scan transfers only matches  
scan rowkey=<cardid>, dealer=<d1234>
  - Allows aggregations on server side (1<sup>st</sup> map already on region server)
  - Some server-side Scan-Filters help already today
  - Java Expression Language?

# *Hbase: missing pieces*

## ➔ Bloomfilter

- Reduce key-lookup time
- Disappeared in 0.20.x, planned to be reanimated in 0.21

## ➔ Hfile persistent internal value index

- Value indexes / Value compression
- Timestamp Index

# Content

- ⇒ What is shared?
- ⇒ Recap RDBMS vs HBase/BigTable
- ⇒ Example: Credit-Card Processing
- ⇒ Storage Patterns
- ⇒ Application Design Proposal
- ⇒ Missing: Call for Features
- ⇒ HbaseExplorer

# Hbaseexplorer: scan



Home CloudInstance List 500/Backup100

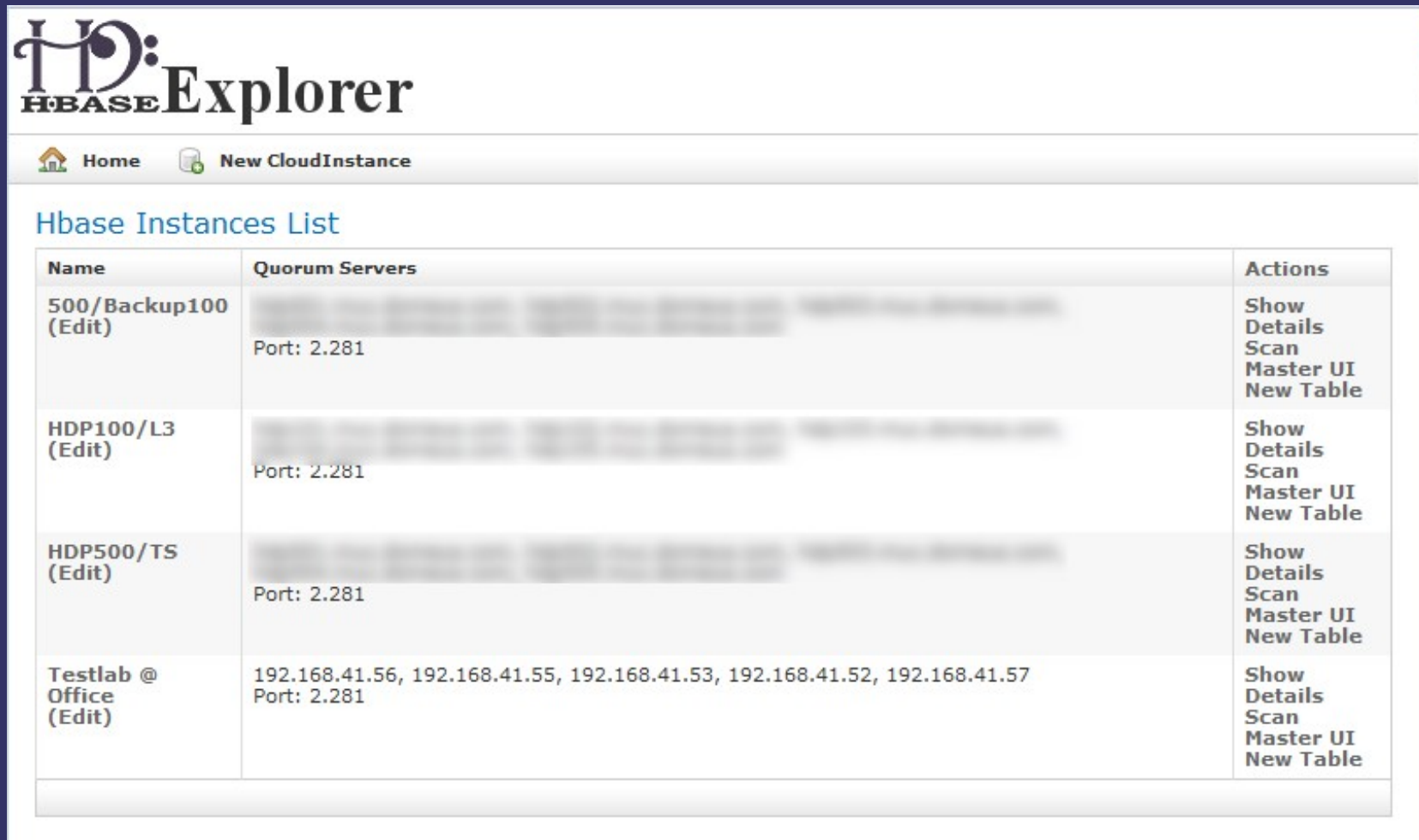
## SCAN on 500/Backup100

Table:  RowKey:   
 Versions:  Rows:

### Scan

RowKey/Timestamp	member	profile	sent	src	tmp	track
<u>2000-2018986274</u> Show all 15 Timstamps		domainmd5: 5c01fa90ff459b9a2a6d60f49143 emailmd5: f29ff826ff92fe67b5929f9ff6d2681 tld: fm ua: Mozilla/5.0 (Windows; U; Windows NT 5.1; pl; rv:1.9.0.17) Gecko/2009122116 Firefox/3.0.17	200148270: 200000833 200152993: 200000833 200165627: 200000833 200168103: 200000833 200171309: 200000833 200174044: 200000833 200177666: 200000833 200179957: 200000833 200183124: 200000833 channel: email messagetype: group rfc822mid: <723060021.168216051267273585946@ecmessenger> transport: ecm-mta-normal-2	app59: app67: host: app68	email: ip: 83.22.	200165627: read 200166790: read 200168103: read 200171309: read 200177666: read
<u>2000-2018986275</u> Hide Timstamps		domainmd5: b1e94d6d1335dc023ee387cfaf46fc1 emailmd5: 25d956984ea01b5c7895cb733852b2ed tld: ru	200148270: 200000833 200152993: 200000833 200165627: 200000833 200168103: 200000833 200171309: 200000833 200174044: 200000833 200177666: 200000833 200179957: 200000833 200183124: 200000833 channel: email messagetype: group rfc822mid: <1599935178.168216081267273585962@ecmessenger> transport: ecm-mta-normal-2	app59: app67: host: app68	email:	
Wed Dec 16 12:20:25 CET 2009 (1260962425185)		domainmd5: b1e94d6d1335dc023ee387cfaf46fc1 tld: ru	200148270: 200000833 channel: email messagetype: group rfc822mid: <1403064828.339200721260962425173@ecmessenger> transport: ecm-mta-normal-2	app59:		
Wed Dec 16 12:28:45 CET 2009 (1260962925663)		emailmd5: 25d956984ea01b5c7895cb733852b2ed				
Thu Dec 24 16:44:44 CET 2009 (1261669484464)			200152993: 200000833 channel: email messagetype: group rfc822mid: <1668206159.537520821261669484451@ecmessenger> transport: ecm-mta-normal-2	app67:		
Mon Jan 25 17:46:21 CET 2010 (1264437981118)			200165627: 200000833 channel: email messagetype: group rfc822mid: <1080769139.1314505771264437981102@ecmessenger> transport: ecm-mta-normal-2	host: app88		


# Hbaseexplorer: cluster setup



The screenshot displays the Hbase Explorer web interface. At the top left is the logo for Hbase Explorer, featuring a stylized 'HD' and the text 'HBASE Explorer'. Below the logo are navigation links for 'Home' and 'New CloudInstance'. The main content area is titled 'Hbase Instances List' and contains a table with four rows of instance information.

Name	Quorum Servers	Actions
500/Backup100 (Edit)	Port: 2.281	Show Details Scan Master UI New Table
HDP100/L3 (Edit)	Port: 2.281	Show Details Scan Master UI New Table
HDP500/TS (Edit)	Port: 2.281	Show Details Scan Master UI New Table
Testlab @ Office (Edit)	192.168.41.56, 192.168.41.55, 192.168.41.53, 192.168.41.52, 192.168.41.57 Port: 2.281	Show Details Scan Master UI New Table

# Hbaseexplorer; Table Definition



Home HbaseSource List 500/Backup100

### Create new Hbase Table on 500/Backup100

Table Name

1	Family Name	<input type="text" value="content"/>		
	Versions:	<input type="text" value="10000"/>	TTL (sec):	<input type="text" value="2147483647"/> 30d: 108000
	<input type="checkbox"/> In Memory	<input checked="" type="checkbox"/> BlockCache	<input type="checkbox"/> Bloomfilter	
	Block Size:	<input type="text" value="65536"/>	Compression:	<input type="text" value="NONE"/>
2	Family Name	<input type="text" value="info"/>		
	Versions:	<input type="text" value="10"/>	TTL (sec):	<input type="text" value="2147483647"/> 30d: 108000
	<input type="checkbox"/> In Memory	<input checked="" type="checkbox"/> BlockCache	<input type="checkbox"/> Bloomfilter	
	Block Size:	<input type="text" value="65536"/>	Compression:	<input type="text" value="GZ"/>



# Hbaseexplorer: statistics



[Home](#)
[HbaseSources](#)
 ->
 [500/Backup100](#)

## HbaseTableStats for Table user on 500/Backup100

Creation	Number of Timestamps	Column Count	Column Size	Value Count	Value Size	Data Size
2010-02-17 21:23:17.584 247.274.340 Rows 263 min for 2041 Regions	2.258.920.782	4.200.423.173	31.962.681.498	11.104.865.871	176.232.229.513	241.798.296.395
Family <b>member</b>	1.063.774.125	322.934.846	2.786.692.983	1.063.774.125	7.974.420.927	13.344.592.678
Family <b>profile</b>	950.519.329	752.862.525	5.012.671.029	950.519.306	35.225.817.464	46.261.388.693
Family <b>sent</b>	6.362.412.873	2.029.965.448	17.993.312.944	6.350.042.133	98.758.669.455	132.991.705.983
Family <b>src</b>	1.433.498.658	579.384.791	2.680.059.875	979.076.044	4.895.380.220	12.210.518.423
Family <b>tmp</b>	1.289.389.037	234.166.446	1.037.121.366	1.289.389.037	27.583.889.900	30.494.342.834
Family <b>track</b>	472.146.926	281.109.117	2.452.823.301	472.065.226	1.794.051.547	6.495.747.784
2010-02-24 23:27:54.618 264.146.383 Rows 377 min for 2441 Regions	2.699.830.594	4.698.326.343	36.093.342.890	12.992.919.855	205.216.982.775	278.896.936.409
Family <b>member</b>	1.262.086.055	361.962.833	3.130.177.327	1.262.086.055	9.457.776.450	15.483.656.441
Family <b>profile</b>	1.047.879.062	816.629.737	5.459.122.758	1.047.879.037	39.813.360.502	51.805.521.156
Family <b>sent</b>	7.600.105.108	2.338.086.033	20.737.297.653	7.585.058.359	117.880.667.276	157.322.653.193
Family <b>src</b>	1.574.709.961	599.968.926	2.762.396.415	1.120.287.347	5.601.436.735	13.163.584.558
Family <b>tmp</b>	1.419.218.194	249.796.169	1.104.132.703	1.419.218.194	30.338.790.165	33.441.292.220
Family <b>track</b>	558.488.541	331.882.645	2.900.216.034	558.390.863	2.124.951.647	7.680.228.841
2010-03-01 15:24:56.592 271.809.632 Rows 296 min for 2663 Regions	2.967.993.962	4.961.247.594	38.344.244.141	13.990.144.242	219.658.514.154	297.692.739.047
Family <b>member</b>	1.380.515.870	383.981.831	3.323.718.075	1.380.515.870	10.342.958.610	16.738.531.333
Family <b>profile</b>	1.096.227.854	845.802.753	5.665.059.833	1.096.227.827	42.292.289.499	54.723.771.356
Family <b>sent</b>	8.327.982.740	2.513.601.808	22.294.754.722	8.311.349.793	128.946.392.432	171.349.961.618
Family <b>src</b>	1.651.126.959	609.059.276	2.798.757.815	1.196.704.345	5.983.521.725	13.654.753.748
Family <b>tmp</b>	1.392.640.438	245.320.540	1.083.045.815	1.392.640.438	29.760.239.231	32.805.849.366
Family <b>track</b>	612.812.420	363.481.386	3.178.907.881	612.705.969	2.333.112.657	8.419.871.626

# *hbaseexplorer*

- ➔ Complements Ruby-Shell
  - Visual Data Representations
  - UI Tools for Table Creation
  - Embedded M/R Jobs for Table Copy or Statistics collection
- ➔ Open Source @ SourceForge
  - Java, WebApp, Grails
  - Coders needed!
- ➔ More info
  - <http://althelies.wordpress.com/hbaseexplorer/>

# eCircle AG

- ⇒ Biggest Direct Email-Marketing Company in Europe
- ⇒ 10 yrs, 200 Employees, now in 6 Countries
- ⇒ Lots of data
  - 100Mio permission Emails / Day
  - Individualized Emails stored, trackings, hosting
  - Privacy challenges → even more data
  - We went through the classic RDBMS scaling story
- ⇒ We hire!
  - Java, UI (JSP, Ajax)



*Thank you!*

[b.schulze@ecircle.com](mailto:b.schulze@ecircle.com)  
[al.lias@gmx.de](mailto:al.lias@gmx.de)