



Spockproxy: a sharding only version of MySQL proxy

Frank Flynn - Database Architect,
Manager of Operations for Spock
Networks

Innovation Everywhere

Spockproxy

- About Spock
- About Spockproxy
- How to Shard your Schema, set up Spockproxy and your database shards. (DBA / SA stuff)
- SQL and the Spockproxy (Developer stuff)
 - what works well
 - what sort of works
 - what does not work

Spock Networks

- Spock is the Leader in Online People Search
- Over 2 billion documents and 500 million people indexed to date
- 12 million unique visitors per month growing at 25% monthly
- MySQL database driven
- Ruby on Rails, active record front end; Perl, Python and more back end.
- We needed: fast queries, large tables and many simulations queries.
- We did not want to change our Application code.

Spockproxy - MySQL proxy

- MySQL Proxy branch - optimized for our needs
 - + Connects to all shards simultaneously (at startup)
 - + Understands id ranges are on specific shards
 - + Will query one or all shards and merge results
 - + Knows where to send reads and writes (all shards, a particular shard or the universal DB)
- No LUA
- Only uses one MySQL account
- Sharding and connection pooling only
- Available on sourceforge

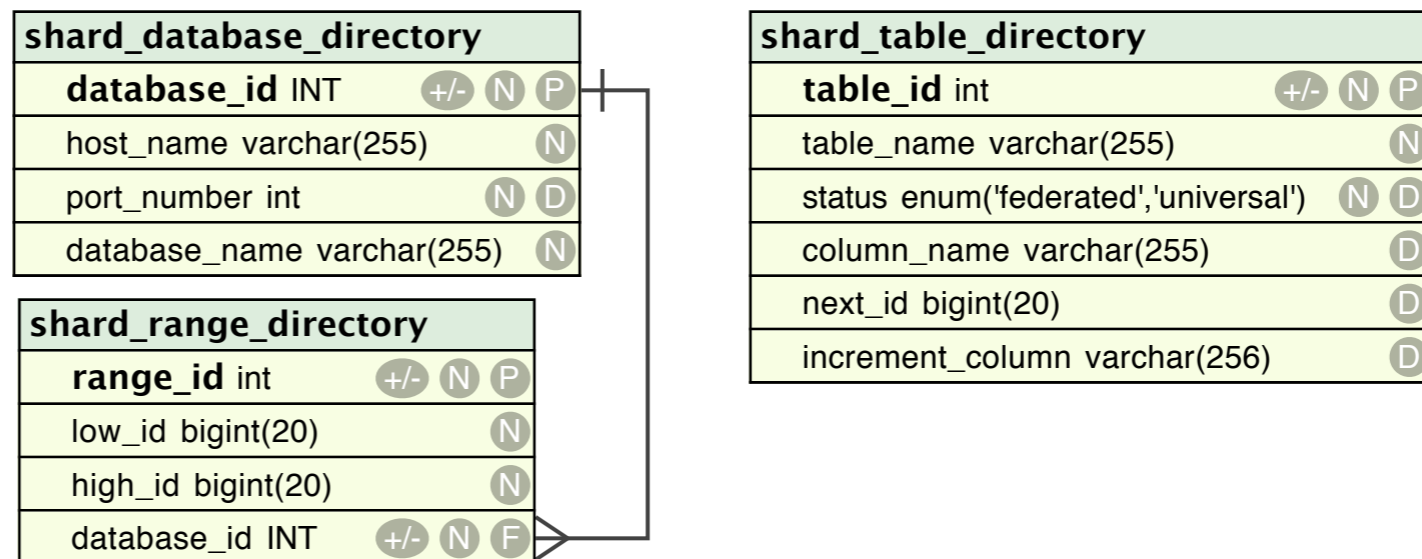
Setup Spockproxy

- Download files from sourceforge
<http://spockproxy.sourceforge.net/>
- build binaries
- Shard your schema
- Set up Universal DB
- Set up Shards



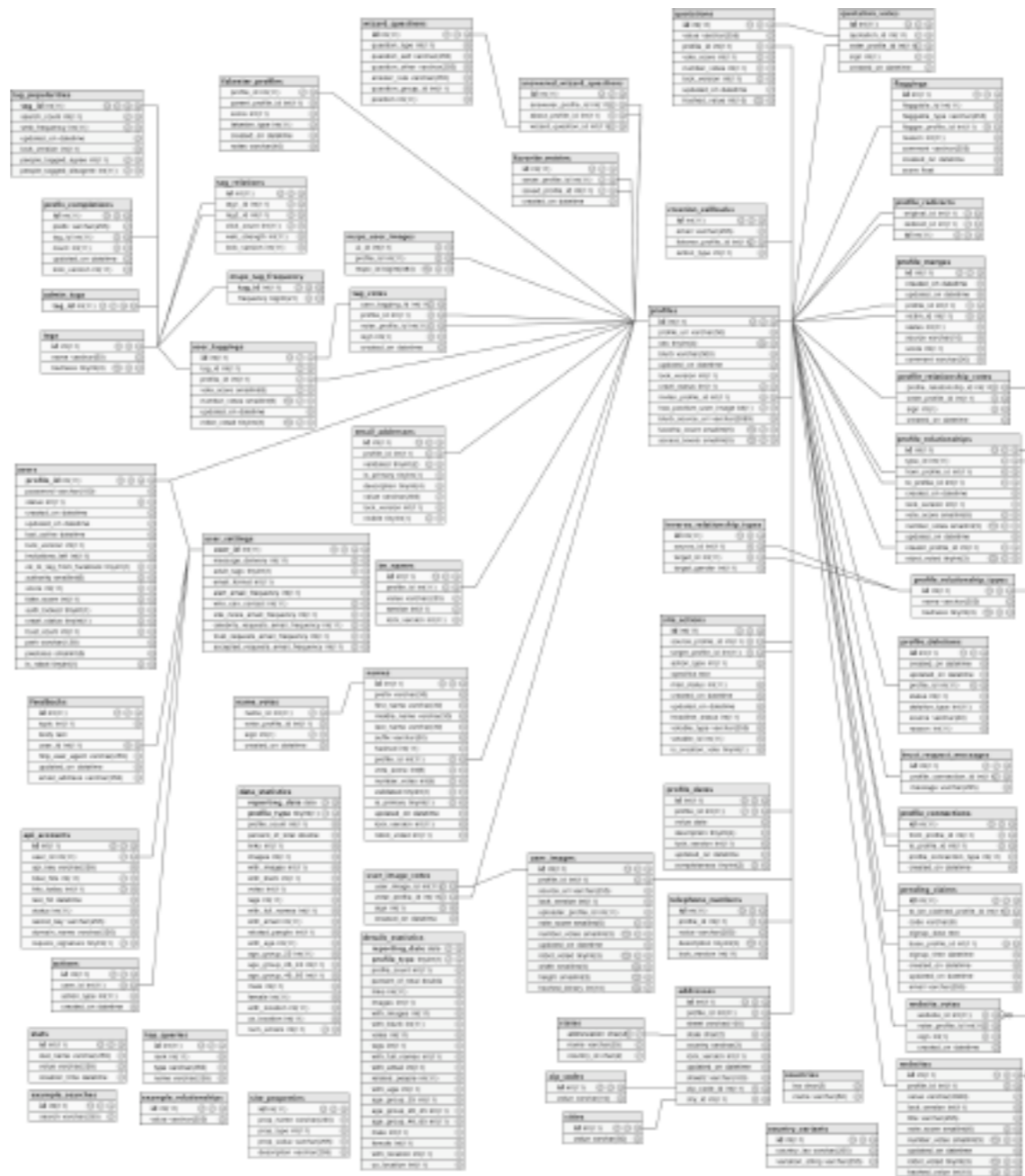
Universal DB

- Is the master database for universal data - data which is common to all shards.
- Contains three directory tables that tell Spockproxy about the shards.



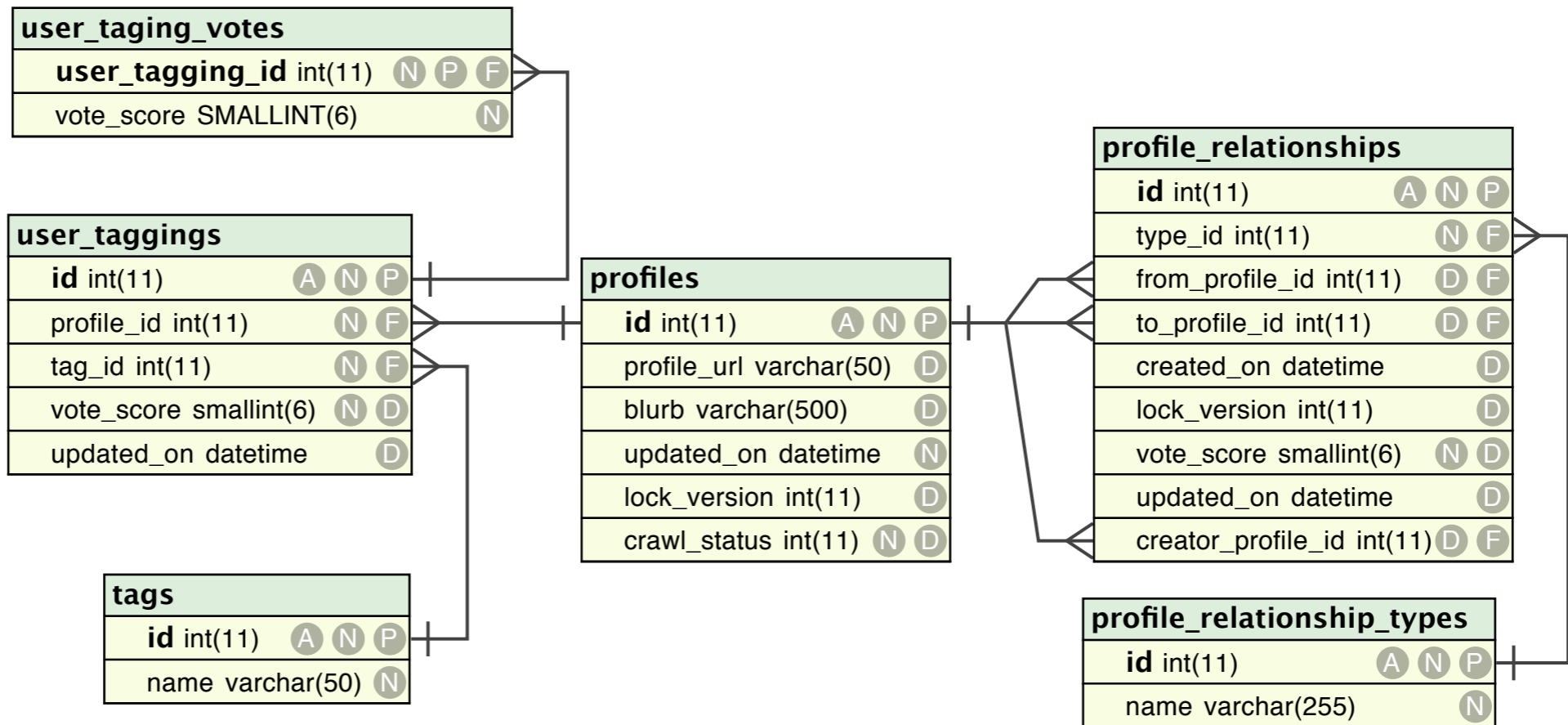
- shard_database_directory
 - list of databases and connection information
- shard_table_directory
 - list of tables are they federated or universal
 - next_id used for get_next_id() automatic increment
- shard_range_directory
 - id ranges for each shard slice

Sharding your Schema



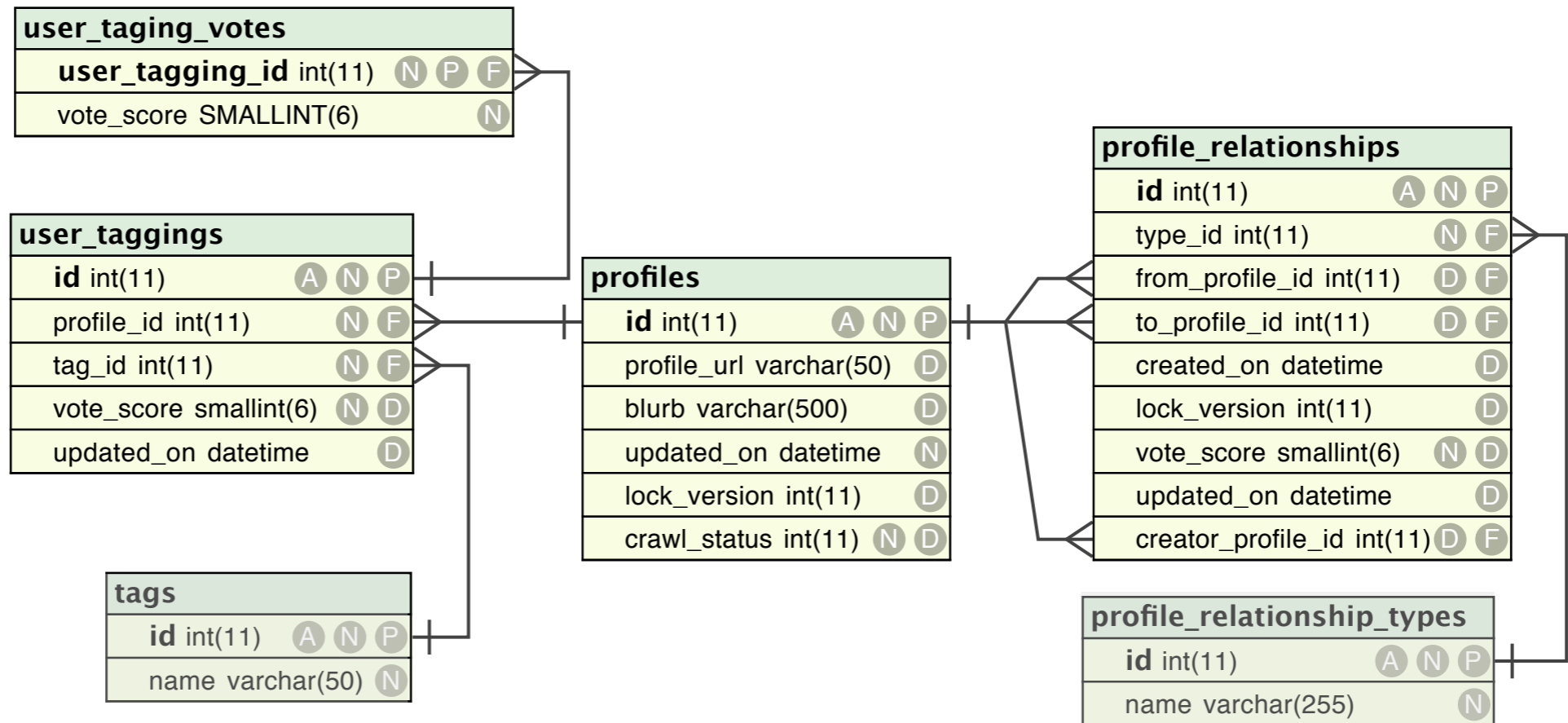
Tables fall into four groups

- 1 - Universal, data with no shard key



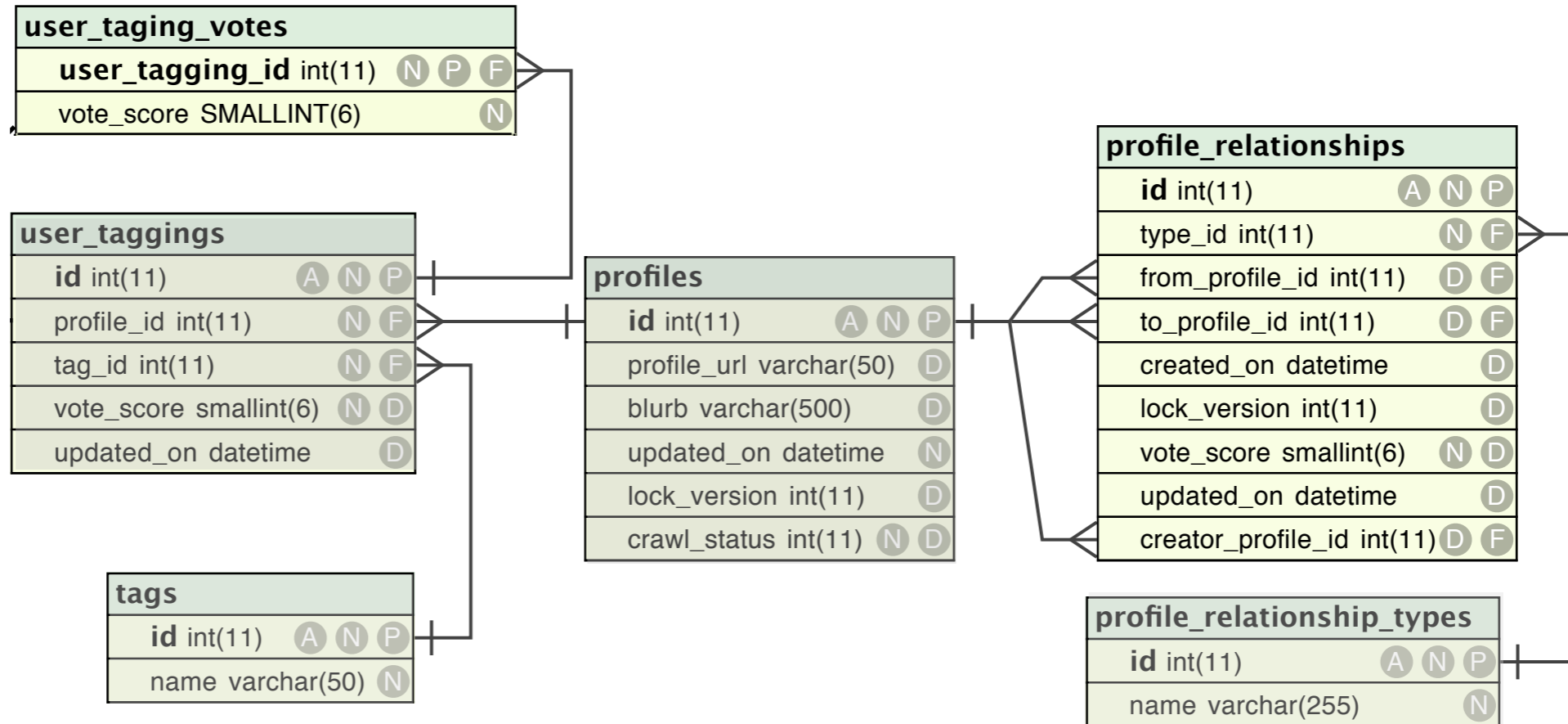
Tables fall into four groups

- 2 - has shard key, data with one shard key



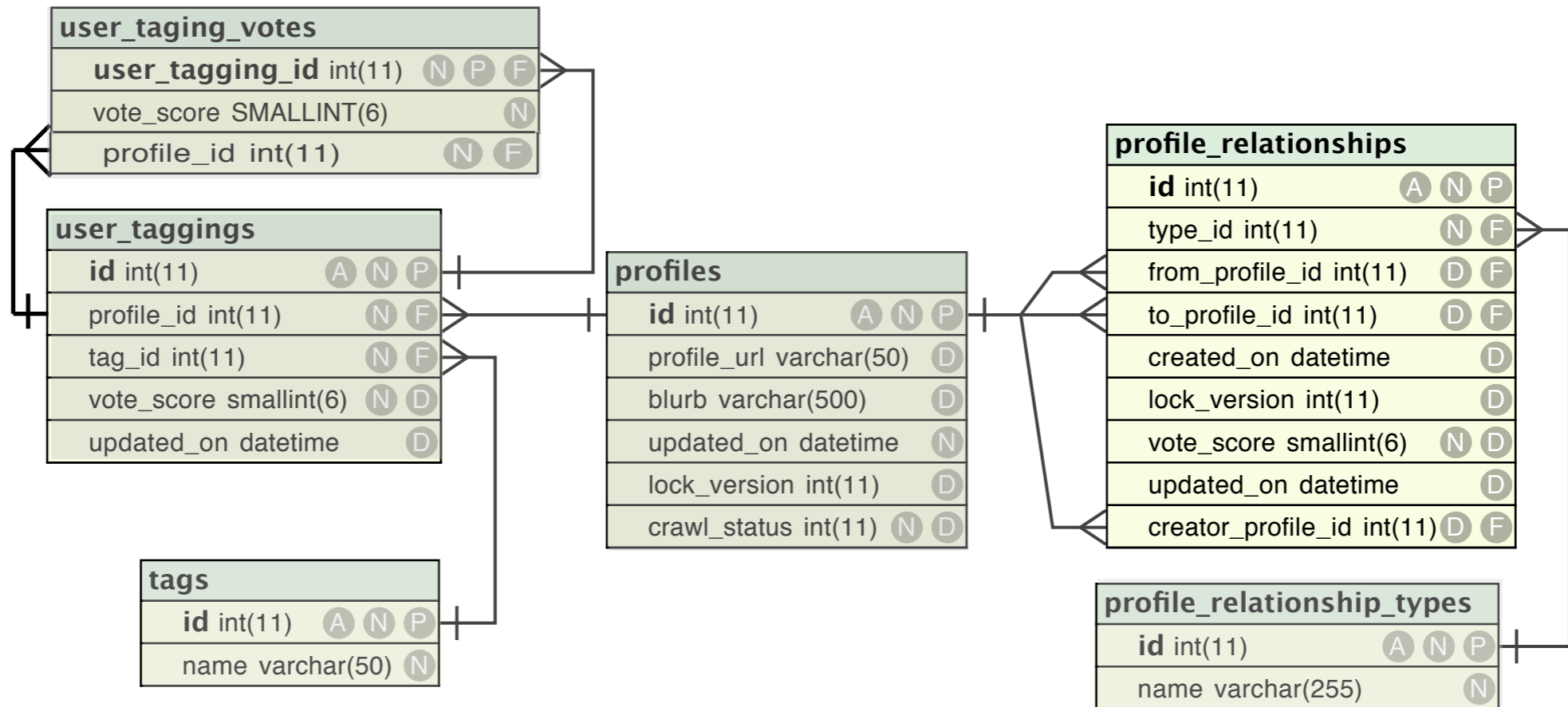
Tables fall into four groups

- 3 - implied shard key



Tables fall into four groups

- 4 - Multiple shard keys available



Categorize your tables

- Make any changes you'll need
- Build the shard_table_directory table
- This query will help:

```
SELECT @m:=0;  
SELECT @m:=ifnull(@m+1,1) AS table_id,  
TABLE_NAME as table_name,  
ELT(MAX(COLUMN_NAME like '%<shard key>%')+1,'universal','federated')  
AS status,  
trim(group_concat(REPEAT(COLUMN_NAME, (COLUMN_NAME like  
'% <shard key>%')) SEPARATOR ' ')) as column_name,  
1 as next_id,  
ELT(MAX(EXTRA like 'auto_increment')+1,',',COLUMN_NAME) AS  
increment_column  
FROM INFORMATION_SCHEMA.COLUMNS WHERE  
TABLE_SCHEMA = 'database name' group by TABLE_NAME;
```

- Open the output and correct as needed
- This will become your shard_table_directory load file.

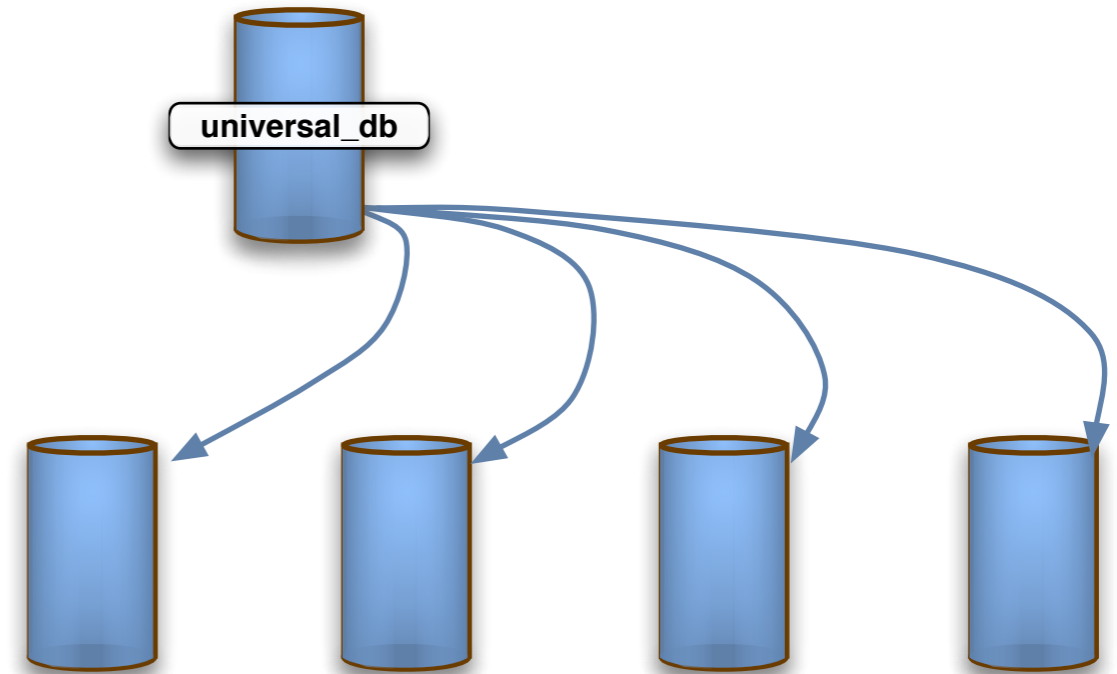
Shard_table_directory load file

- Is the Status (federated, universal) correct?
- Did we pick the correct shard key?

table_id	table_name	status	column_name	next_id	increment_column
1	aux_types_limited	universal		1	id
2	aux_type_keys	universal		1	key_id
3	categories	universal		1	id
4	email_address	federated	profile_id	1	id
5	profiles	universal		1	id
6	profile_relationships	federated	from_profile_id, creator_profile_id, to_profile_id	1	id
7	profile_relationships_types	universal		1	
8	tags	universal		1	
9	user_taggings	federated	profile_id	1	id
10	user_tagging_votes	universal		1	id
11	user_images	federated	profile_id	1	id

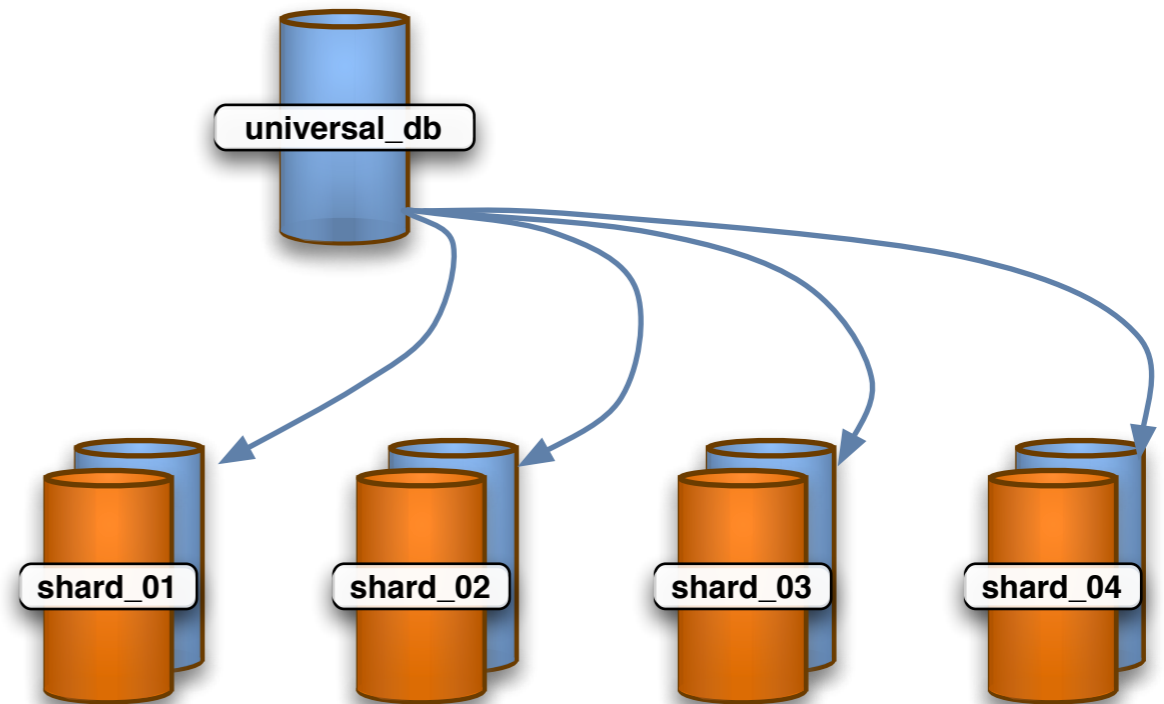
- **column_name** is the shard key
- Make corrections here and if necessary to your schema

Set up databases: Universal DB



- Directory tables
- Replicate a copy to each shard server
- Load all “universal” tables

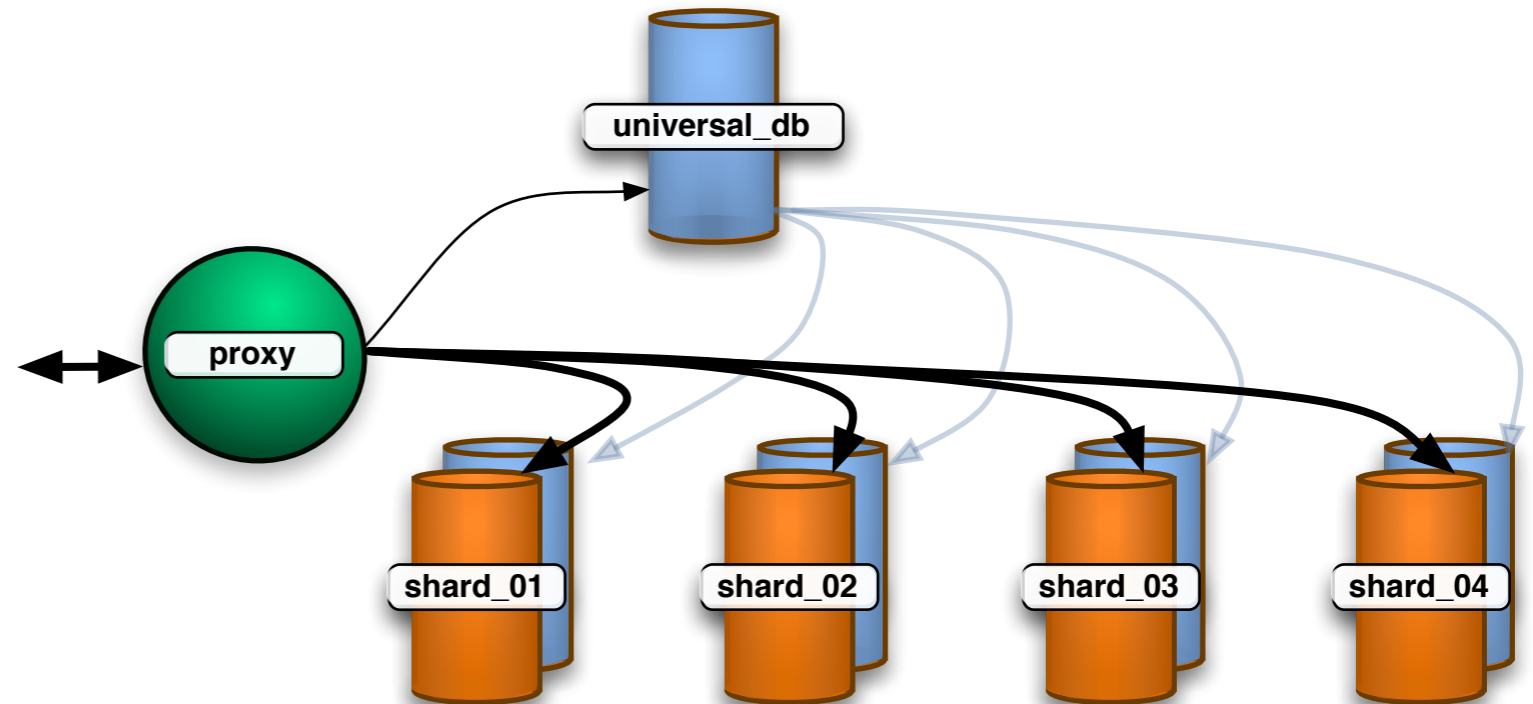
Set up databases: shards



- All sharded tables
- A view for every table in the local Universal DB

Set up Spockproxy server

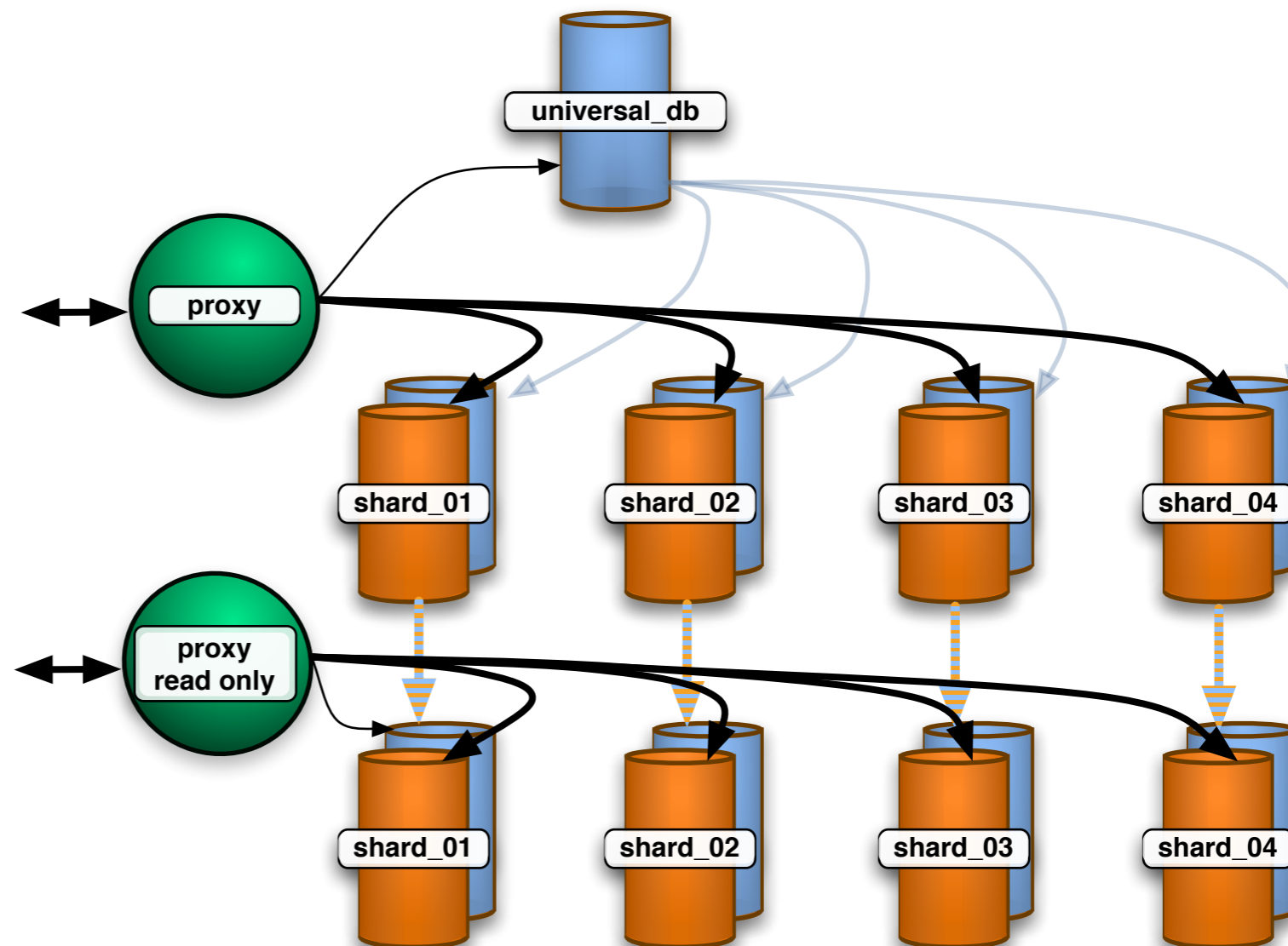
Proxy is ready for clients



- Config file (or command line) tells proxy:
 - how to connect to universal DB
 - login account information
- Universal DB tells proxy about the rest of the shards and tables



Set up replication



- Replicate the shard servers but not the shard_database_directory table (put the replicas in here)
- Set up a second proxy

Move the Data

- Dump and load the schemas (one for universal and one for the shards).
- Dump the data in the ranges for the shards.
- Load data into each specific shard.
- Create a view in each shard to every table in it's local universal replica.
- There are handy scripts that will read the `shard_..._directory` tables and create dump and load scripts and create view scripts for you.

Spockproxy is ready

- SQL commands are largely the same but there are some important differences.
- Commands are sent to one or all servers as appropriate.
- Servers cannot see each others data.
- Joins by the shard key work.
- Joins that cross shard boundaries do not work.

- Take a look at specific commands:



SELECT

- SELECT is sent to one or all Shards, never to the Universal DB. Unless inside of a transaction.
- SELECT WHERE <shard key> = <value> is sent to one Shard.
- SELECT without a shard key is sent to all shards and results are merged.
- Sub-queries are NOT supported - BUT...
- IN (list) IS supported
.... WHERE col IN (10, 27, 85, 943, 1034)
- Remember you cannot join across shards

Transactions

- Work within each shard but be careful, you can lock up everything.
- BEGIN TRANSACTION is transmitted to all shards and universal DB when you send it to the proxy.
- Those connections are yours until you COMMIT or ROLL BACK
- If one shard fails during a transaction - the other shards are NOT rolled back (that's up to you).



INSERT INTO

- Only single row inserts through proxy
- Sharded data **MUST** have shard key
- Universal tables are inserted into universal DB and replicated (possible delay before they show up in a SELECT statement)
- No nested queries:

```
INSERT INTO tbl_name  
SELECT FROM other_tbl_name
```
- auto_increment is different

auto_increment

- Problem:
auto_increment works at the server
 - primary key overlap between shards
 - the proxy does not know where to put data without the shard key
 - last_insert_id() is meaningless with pooled connections
- One Solution:
SELECT get_next_id(<table name>, <number of id's>)
 - must be run before insert if you want to know the id
 - if you don't care the proxy will do this for you
 - can give you many id's at once for bulk loading
 - you can call it directly in the universal database



Bulk Loading

- Problem:
Proxy must examine every row being inserted
 - INSERT INTO tbl_name (a,b,c) VALUES(1,2,3), (4,5,6),(7,8,9);
 - LOAD DATA INFILE 'file_name' INTO TABLE tbl_name
Are NOT supported
- Two Solutions:
 - Load the shards directly:
best performance
more code for you to write
 - INSERT one row at a time
easy but slow

UPDATE and DELETE

- Never update the Shard Key - use delete and insert to keep data in the correct shard
- Sub-queries are NOT supported.
- IN (list) IS supported.
- With the Shard Key your statement is sent to one shard; without the Shard Key your statement sent to all of them.



Stored Routines

- Limited Support
- Sent to all Shards
-except for `get_next_id()`
- Results are merged
- Remember: they are run within each Shard so `SELECT max(id) FROM ...` within a routine can be different in each shard.

```
mysql> SELECT CONNECTION_ID();
+-----+
| CONNECTION_ID() |
+-----+
|          3007   |
|          2936   |
|          2961   |
|          2960   |
+-----+
4 rows in set (0.00 sec)
```



Supported

- Aggregate functions: min(), max(), count(*), sum() with or without GROUP BY
- GROUP BY works within a shard only, then the results are merged. You might be able to correct this in your application, for example:

```
SELECT count(*), sum(col)...
```

then compute the average yourself.
- ORDER BY - will be sorted in the shard then again in the proxy during the merge
- LIMIT and OFFSET - work as expected however this is done in the proxy server and requires the proxy retrieve all the data then apply LIMIT and OFFSET

NOT Supported

- USE DATABASE
- ALTER, CREATE, DROP, ... (TABLE, INDEX, PROCEDURE, FUNCTION, ...)

- INFILE, OUTFILE*

*this will work but consider it will be send to all shards, you could then collect all the files and merge them yourself.



Spockproxy Questions:

- Important contact information:
- <http://spockproxy.sourceforge.net/>
source code, email list, demo db, helpful scripts
- <http://www.frankf.us/>
my blog - database and other things when I have time
- frank@frankf.us - ask me interesting questions and I might write a blog about it.