Mailchimp Scale: A MySQL Perspective

John Scott
Mailchimp
What is Mailchimp’s secret sauce?

Hint: It’s not much of a secret.
Focus on the small business

“Empowering the Underdog”
“We give marketers production-ready software designed to help them grow...”

Mailchimp Engineering Mission Statement
https://mailchimp.com/culture/how-our-engineering-team-found-its-mission-statement/
WE SUCCEED THROUGH
TOGETHERNESS,
MOMENTUM,
AND PRAGMATISM.
Another way to say it

“We SCALE through togetherness, momentum, and pragmatism.”
Old Mentality: The 3 Disciplines of Data Administration

- OPS / KTLO
- Support
- Performance
Old Mentality: The 3 Disciplines of Data Administration

- OPS / KTLO
- Support
- Performance

“I’m a DevOps DBA”
Old Mentality: The 3 Disciplines of Data Administration

- OPS / KTLO
- Support
- Performance
Old Mentality: The 3 Disciplines of Data Administration

- OPS / KTLO
- Support
- Performance

“I help other departments work with databases”
Old Mentality: The 3 Disciplines of Data Administration

- OPS / KTLO
- Support
- Performance
Old Mentality: The 3 Disciplines of Data Administration

- OPS / KTLO
- Support
- Performance

“over the fence”
New Mentality:

“Ops is product”
“If you improve database performance resulting in 10% reduction in churn, you would create an additional <big revenue number>.”
Ops is Product

“Developer Enablement”

New paradigm “looking at ops through the lens of product” -- Tyler Treat

- [https://www.youtube.com/watch?v=JUy3GYkPfo](https://www.youtube.com/watch?v=JUy3GYkPfo)

OR in the case of Mailchimp, **ops actually developing software**, too.
Developer Enablement
Product Enablement

In most organizations “Product enablement” is a sales term with the “four Ps”

- Positioning
- Pitch
- Play
- Program
Developer Enablement
Product Enablement

1000 employees

350+ engineers

0 salespeople
Mailchimp “Board Room”
LISTEN HARD

CHANGE FAST
Sounds great.
But what does that mean for a database engineer?
#togetherness in action

MySQL log analysis based on pt-query-digest and Elasticsearch / Kibana resulted in a Top 20 table activity graph.
End of story?

“Toss it over the wall.”
“Not my problem.”
“I don’t have commit rights.”
This is Mailchimp Engineering

“We succeed through togetherness, Momentum, and Pragmatism”
We identified an N+1 pattern and fixed it, together.
But wait....
What was the impact to the user experience?
billion queries per week

thousand unique query fingerprints

Instances of mysql
Old Mentality: Effective Slow Query Log Analysis Across The Infrastructure FTW!

“Query Macroeconomics”
https://johnscott.net/2018/08/03/query-macroeconomics/

- Prioritize query fixes by how much DB capacity you get back
  - MySQL not stressed with contention equals what?
    - A pretty innodb status?
    - Nice looking graphs?
Old Mentality: Effective Slow Query Log Analysis Across The Infrastructure FTW!

“Query Macroeconomics”
https://johnscott.net/2018/08/03/query-macroeconomics/

- Prioritize query fixes by how much DB capacity you get back
  - MySQL not stressed with contention equals what?
    - A pretty innodb status?
    - Nice looking graphs?
Old Mentality: Effective Slow Query Log Analysis Across The Infrastructure FTW!

“Query Macroeconomics”
https://johnscott.net/2018/08/03/query-macroeconomics/

- Prioritize query fixes by how much DB capacity you get back
  - MySQL not stressed with contention equals what?
    - A pretty innodb status?
    - Nice looking graphs?
Old Mentality: Effective Slow Query Log Analysis Across The Infrastructure FTW!

“Query Macroeconomics”
https://johnscott.net/2018/08/03/query-macroeconomics/

- Prioritize query fixes by how much DB capacity you get back
  - MySQL not stressed with contention equals what?
    - A pretty innodb status?
    - Nice looking graphs?
“Ops is Product”

Can a DBE team improve performance and capacity in a silo?
“Ops is Product”

Can a DBE team improve performance and capacity in a silo?
“Ops is Product”

Can a DBE team reduce churn by 10% in a silo?
“Ops is Product”

Can a DBE team reduce churn by 10% in a silo?
We identified an N+1 pattern and fixed it, together.
We enriched the sessions with context about the user, how the session was accessed and other pertinent information. This context was sent to the slow query logs and included in the session data.
This new session analysis led to more improvements, more togetherness, and a better experience for our customers.
How Mailchimp Avoids Silo #togetherness

- All engineers have code repository access
- Transparent, pragmatic standards
- Empowering each other to suggest and make changes outside of core role
- Everyone is on Slack
- Multi-Disciplinary approach
  - We don’t make infrastructure decisions alone as DBEs
  - DBEs are not on-call alone
  - DBEs contribute code
How Mailchimp Avoids Silo #togetherness

- All engineers have code repository access
- Transparent, pragmatic standards
- Empowering each other to suggest and make changes outside of core role
- Everyone is on Slack
- Multi-Disciplinary approach
  - We don’t make infrastructure decisions alone as DBEs
  - DBEs are not on-call alone
  - DBEs contribute code
How Mailchimp Avoids Silo #togetherness

- All engineers have code repository access
- Transparent, pragmatic standards
- Empowering each other to suggest and make changes outside of core role
- Everyone is on Slack
- Multi-Disciplinary approach
  - We don’t make infrastructure decisions alone as DBEs
  - DBEs are not on-call alone
  - DBEs contribute code
How Mailchimp Avoids Silo #togetherness

- All engineers have code repository access
- Transparent, pragmatic standards
- Empowering each other to suggest and make changes outside of core role
- Everyone is on Slack
- Multi-Disciplinary approach
  - We don’t make infrastructure decisions alone as DBEs
  - DBEs are not on-call alone
  - DBEs contribute code
How Mailchimp Avoids Silo #togetherness

- All engineers have code repository access
- Transparent, pragmatic standards
- Empowering each other to suggest and make changes outside of core role
- Everyone is on Slack

- Multi-Disciplinary approach
  - We don’t make infrastructure decisions alone as DBEs
  - DBEs are not on-call alone
  - DBEs contribute code
DBE code contributions (current)

- Fixing bad queries
- Code /process improvement
- Data residence change
- Participation in green field projects
- Compliance
- Wherever we find we are needed / useful
DBE code contributions (current)

- Fixing bad queries
- *Code/process improvement*
  - Data residence change
  - Participation in green field projects
  - Compliance
  - Wherever we find we are needed / useful
DBE code contributions (current)

- Fixing bad queries
- Code /process improvement
- Data residence change
- Participation in green field projects
- Compliance
- Wherever we find we are needed / useful
DBE code contributions (current)

- Fixing bad queries
- Code /process improvement
- Data residence change
- Participation in green field projects
- Compliance
- Wherever we find we are needed / useful
DBE code contributions (current)

- Fixing bad queries
- Code/process improvement
- Data residence change
- Participation in green field projects
- Compliance
- Wherever we find we are needed / useful
DBE code contributions (current)

- Fixing bad queries
- Code /process improvement
- Data residence change
- Participation in green field projects
- Compliance
- Wherever we find we are needed / useful
“The Boring Part”

A few technical details about Mailchimp and the simplistic way we run MySQL
MySQL Instances at Mailchimp
MySQL Instances at Mailchimp
Infrastructure Evolution

Instances used to be standalone. Each on its own server on spinny disk, but not anymore.

New way: multi-instance per server, “pool”

ODD NUMBER SERVER

Server db1.pool

ADJACENT EVEN NUMBER SERVER

Server db2.pool

MySQL replication
Infrastructure Evolution

Average density:

2200 (instances) / 725 (hosts)
(3 instances per host and climbing)
How we got to 2200 instances easily

Automated user moves:

Add instances,

adjust configs,

users get rebalanced across new instances
Infrastructure Evolution

- **Old way (instance per server)**
  - Ex: HP Gen 8, 32 core, 48GB RAM, 512G RAID 10 (spinner)
  - Instance split case: “bufferpool calculated by disk usage”

- **New(er) way: multi-instance servers**
  - Ex: HP Gen 10, 56 core, 256GB RAM, 6T (NVME)
  - Up to 8 instances
  - Split case “divide bufferpool evenly”

- **Both single tenant and multi-tenant schemata** (hundreds of thousands of schemata, millions of innodb containers)
“Standing on the shoulders of giants”
Tooling (3rd party)

- Infrastructure automation (puppet)
- Decent monitoring, alerting and trending
  - Zabbix
  - OpsGenie
  - Prometheus
  - Grafana
  - ELK

Administered in collaboration with other specialized teams

Using open source templating in some cases (PMM dashboards)
Tooling (home grown)
DCM or “Data Center Manager”

- Add/drop instances without logging into servers
- Use the list function to return lists of servers within other scripts
- Automatic configuration (interoperation with puppet)
  - Backups
  - Replication
  - Virtual IP
Great Support
Pragmatism

MySQL Orchestration Technology:

- Past: MMM
  - Present: home grown
- Future: Orchestrator?
Pragmatism

MySQL Orchestration Technology:

- Past: MMM
- Present: home grown
- Future: Orchestrator?
Pragmatism

MySQL Orchestration Technology:

- Past: MMM
- Present: home grown
- Future: **Orchestrator**?
Pragmatism

MySQL Orchestration Technology:

- Past: MMM
- Present: home grown

- Future: Orchestrator?
- MHA
Pragmatism: Why MHA?

- Orchestrator requires its own infrastructure.
  - its own database pair
  - its own web server
- We already have a kubernetes cluster.
- MHA docker containers can be managed through DCM, github and existing PR/merge process.
- Easy to deploy, easy to monitor with existing infrastructure.
Pragmatism: Why MHA?

- Orchestrator requires its own infrastructure.
  - its own database pair
  - its own web server
- We already have a Kubernetes cluster.
- MHA docker containers can be managed through DCM, github and existing PR/merge process.
- Easy to deploy, easy to monitor with existing infrastructure.
Pragmatism: Why MHA?

- Orchestrator requires its own infrastructure.
  - its own database pair
  - its own web server
- We already have a kubernetes cluster.
- MHA docker containers can be managed through DCM, github and existing PR/merge process.
- Easy to deploy, easy to monitor with existing infrastructure.
Pragmatism: Why MHA?

- Orchestrator requires its own infrastructure.
  - its own database pair
  - its own web server
- We already have a Kubernetes cluster.
- MHA docker containers can be managed through DCM, Github and existing PR/merge process.
- Easy to deploy, easy to monitor with existing infrastructure.
Pragmatism: Why MHA?

- Orchestrator requires its own infrastructure.
  - its own database pair
  - its own web server
- We already have a kubernetes cluster.
- MHA docker containers can be managed through DCM, github and existing PR/merge process
- Easy to deploy, easy to monitor with existing infrastructure
Old virtual IP management

- Puppet pushes instance configs to centralized daemon.
- The “mysql-vip” homegrown daemon checks DB availability & replication lag.
- The daemon SSHs to db servers to move VIP.
  - on demand or
  - in the event of issue
- Downstream replicas are not managed.
- The read_only flag is not set on off-master.
MHA Deployment

- The MHA repository in git contains:
  - Docker entrypoint
  - MHA itself
  - Supporting scripts to avoid split brain
  - Container definition per instance generated via script against configuration files
- Changes peer reviewed in github
- Downstream replicas managed
- The read_only flag is set.
  - supports ProxySQL in the future
What’s Next

- ProxySQL
- Cloud
- Many other team-enabled optimizations
  - Data tenancy
  - Legacy replacement
  - New features
What’s Next

- ProxySQL
- Cloud
- Many other team-enabled optimizations
  - Data tenancy
  - Legacy replacement
  - New features
What’s Next

- ProxySQL
- Cloud
- Many other team-enabled optimizations
  - Data tenancy
  - Legacy replacement
  - New features
What’s Next

- ProxySQL
- Cloud
- Many other team-enabled optimizations
  - Data tenancy
  - Legacy replacement
  - New features
Don’t be afraid to seek #togetherness in your own company.

How can you make OPS=PRODUCT in your org?

Pragmatism vs newest tech.

Feel empowered to fix what is within your power to change.

Inspire others. Each one teach one, each one reach one.
DBE Empowerment & Product Enablement

Don’t be afraid to seek #togetherness in your own company.

How can you make OPS=PRODUCT in your org?

Pragmatism vs newest tech.

Feel empowered to fix what is within your power to change.

Inspire others. Each one teach one, each one reach one.
DBE Empowerment & Product Enablement

Don’t be afraid to seek #togetherness in your own company.

How can you make OPS=PRODUCT in your org?

Pragmatism vs newest tech.

Feel empowered to fix what is within your power to change.

Inspire others. Each one teach one, each one reach one.
Don’t be afraid to seek #togetherness in your own company.

How can you make OPS=PRODUCT in your org?

Pragmatism vs newest tech

Feel empowered to fix what is within your power to change.

Inspire others. Each one teach one, each one reach one.
Don’t be afraid to seek #togetherness in your own company.

How can you make OPS=PRODUCT in your org?

Pragmatism vs newest tech.

Feel empowered to fix what is within your power to change.

Inspire others. Each one teach one, each one reach one.
Thank you.