GETTING DATA OUT OF DATABASES:

A SURPRISINGLY TRICKY PROBLEM!

MARTIN KLEPPMANN
@martinkl
UNIVERSITY OF CAMBRIDGE
Designing Data-Intensive Applications

Martin Kleppmann

@martinkkl

dataintensive.net
Web app

DB
NO ONE O B TO RULE THEM ALL
"Do one thing and do it well" vs. transaction processing, analytics, full-text search, replication, spatial index, materialized views, OLAP, machine learning, graph index, monitoring, time-series data, auditing, caching, notifications.
SAME DATA in DIFFERENT FORM
SAME DATA in DIFFERENT FORM

Denormalisation

Caching

Indexes

Aggregations
set X = A

ok

ok
set $X = A$

set $X = A$

set $X = B$

set $X = B$
set \( X = A \) set \( X = A \)

set \( X = B \) set \( X = B \)
STOP DOING THIS.
Our Challenge:

Data Integration

"Making sure the data ends up in all the right places"
yesterday
Transaction log: Append

1 2 3 4 5 6 7 8 9 10

- yesterday
- last hour
Transaction log

Append

1 2 3 4 5 6 7 8 9 10 11 12

...}

yesterday

last hour

last minute
Log file

Append

↑
tail -f
like UNIX pipes
but for DISTRIBUTED DATA
mysql | elasticsearch
like UNIX pipes
but for DISTRIBUTED DATA
mysql | elasticsearch
tail -f access.log | join <(mongodb) | recommendations | sendmail
All Aboard the Databus!

Linkedin’s Scalable Consistent Change Data Capture Platform

Shirshanka Das, Chavdar Botev, Kapil Surlaker, Bhaskar Ghosh
Balaji Varadarajan, Sunil Nagaraj, David Zhang, Lei Gao
Jemiah Westerman, Phanindra Ganti, Boris Shkolnik, Sajid Topiwala
Alexander Pachev, Naveen Somasundaram and Subbu Subramaniam

LinkedIn
Mountain View, CA, USA
{sdas,cbotev,ksurlaker,bghosh,bvaradar,snagaraj,dzhang, lgao, jwesterman}@linkedin.com
{pganti,bshkolnik,stopiwala,apachev,nsomasundaram,ssubramaniam}@linkedin.com

3rd ACM Symposium on Cloud Computing, San Jose, Oct 2012
USING CHANGE CAPTURE

Person submit data to Web app, which writes to DB.
USING CHANGE CAPTURE

Web app

Kafka (with log compaction)

write

extract changes

append to log
Using Change Capture

- User submits data to Web app.
- Web app writes to DB.
- Extract changes from DB.
- Kafka (with log compaction) stores changes.
- Append to log.
- Index.
Using Change Capture

- A user submits data to a web app.
- The data is written to a database (DB).
- Extract changes from the database.
- Kafka is used with log compaction.
- Changes are appended to the log.
- Data is stored in an index and cache.
Using change capture

1. User submits data
2. Data is handled by the Web app
3. Data is written to the DB
4. Changes are extracted
5. Data is routed to Kafka
6. Kafka processes the data with log compaction
7. Data is then directed to Index, Cache, and HDFS
8. Consumers independently apply writes
Using Change Capture

1. User submits data to Web app.
2. Data is written to DB.
3. Extract changes from DB.
5. Kafka passes data to Index, Cache, and HDFS.
6. Monitoring and Samza handle data processing.
DATA STREAMS - FRESHLY BOTTLED AT SOURCE
A Stream Data Platform powered by Apache Kafka
Web app

Postgres

(read/write
(as usual))
Kafka

Apache Software Foundation
Kafka message broker
PRODUCERS

- Web app
- Mobile app
- Email
- ...

Kafka message broker

user activity

events

bounces
Log file

Append

1 2 3 4 5 6 7 8 9 10 11 12

↑

tail -f
stream

oldest events    most recent events
stream

new events added here

← oldest events

most recent events →
stream

new events added here

oldest events

most recent events

real-time consumer position (close to head of stream)
oldest events

append

time

Index
Kafka changelog compaction

A 42  B 21  A 44  C 12  A 45

↓

B 21  C 12  A 45
stream new events added here

oldest events

most recent events

complete history
(-using compaction to collect garbage)
New index (starts off empty)
New index (starts off empty)
like UNIX pipes
but for DISTRIBUTED DATA
mysql | elasticsearch
tail -f access.log | join <(mongodb) | recommendations | sendmail
USING CHANGE CAPTURE

- A person submits data to a Web app, which writes to a DB.
- Changes are extracted and added to Kafka.
- Kafka (with log compaction) is used to append to the log.
- Extracted changes are then applied to Index, Cache, and HDFS by independent consumers.

consumers independently apply writes
Further reading


Free copies in the lunch break!

Discount code: TS2015

dataintensive.net

Bottled Water

[github.com](http://github.com)/confluentinc/bottledwater-pg