ORACLE DevLive | Level Up | MySQL Summit

Reliable Operations and Rapid Development with MySQL

- **Nicolai Plum – Senior Database Engineer**
- Booking.com

Booking.com

Reliable Operations and Rapid Development with MySQL

Nicolai Plum
MySQL Database Engineering - Booking.com

Booking.com

Booking.com

100M
monthly active app users

232M+
verified guest reviews and
24/7
customer service
in **44**
languages and dialects

Since 2010, Booking.com has welcomed
4.5B+
guest arrivals

28M
total reported listings worldwide

6.6M
options in homes, apartments and other unique places to stay

140 offices in **70** countries over
5,000 employees in Amsterdam

30
different types of places to stay,
including homes, apartments, B&Bs, hostels, farm stays, bungalows, even boats, igloos and treehouses

155,000
destinations around the world

Car hire available in **140+** countries and pre-booked taxis in over **500** cities across **120+** countries

B.

MySQL for 18+ years: What?

- Core transaction processing
- FinTech, Payments & Billing
- Front-end content
- Partner and Customer Support
- Internal tools and controlplanes

MySQL: Why?

- Good Replication Model (scale-out reads)
- Fast (Connect, Point select, Range select)
- Data durability (InnoDB is extremely solid)
- Easy to manage
- Open license

MySQL: How?

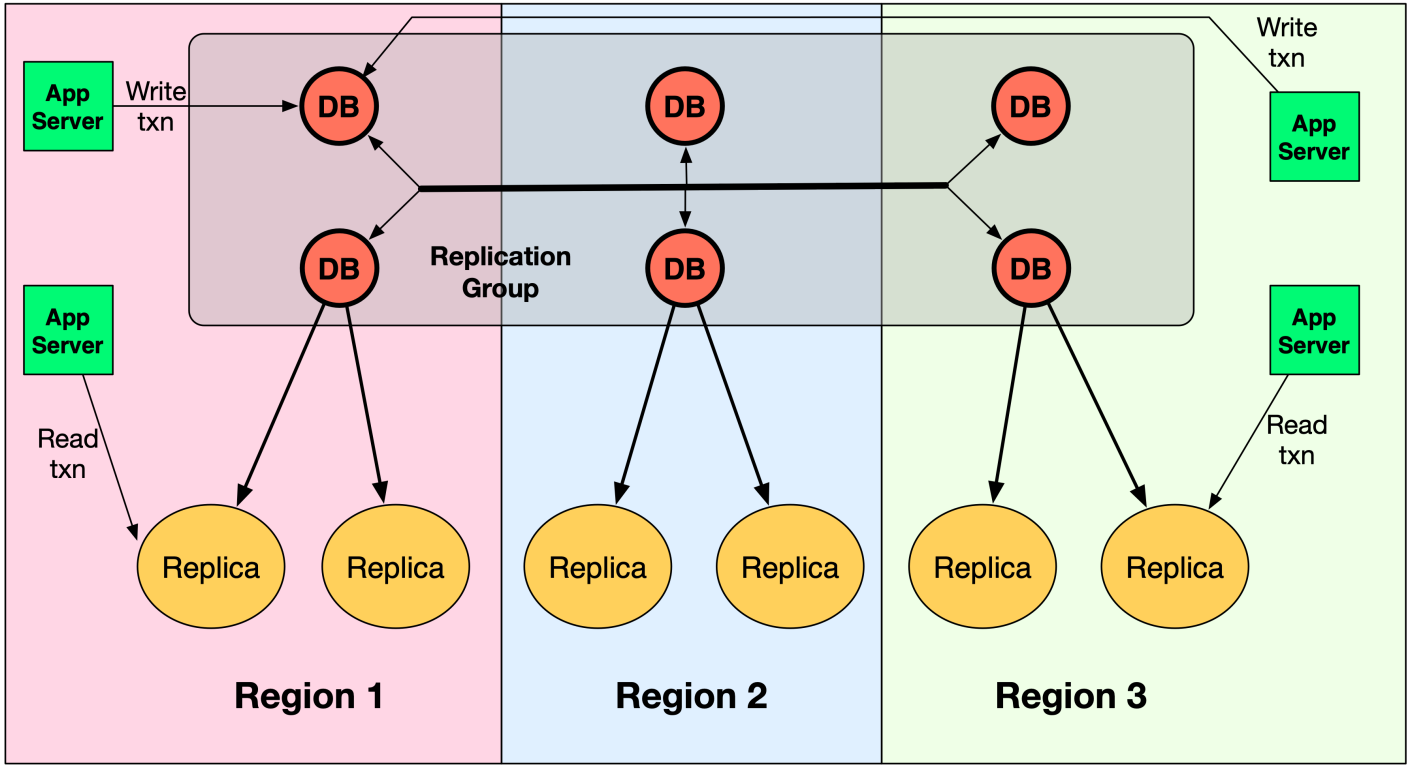
Replace **Break/Fix**
with
Preventative Maintenance

Booking.com

Relentless automation

- Monitoring, connected to self-healing
- Auto provisioning (CLONE plugin)
- Auto maintenance (upgrade, patch, replace)
- Auto grants & service discovery group config
- Autoscaling replica count

MySQL Group Replication



Booking.com

Group Replication

- Single Primary
- Paxos Single Leader
- Latest version (also of group protocol)
- Across failure domain (region or AZ)
- Minimise Storage & Network Latency

What about the data?

- Schema changes must be fast
  - Avoid DBAs blocking developers
- ORMs want to manage schema
  - Hibernate, Flyway, Django
- Developers are not DB Architects
  - … and may not want to be

Guidance + Tools

- Design guidance (docs & training)
  - Must be approachable and digestible
- Automated interface – GUI or CI/CD
- Must cover **all** cases, or you get outages
  - pt-osc or similar is necessary because ONLINE DDL blocks replication
- Online DDL is not Instant DDL
  - Frequent failure mode of “do it all” ORMs.

Better ALTER

- You run tests, but do you report DB changes explicitly?
- Copy table, or schema, to blank database
- Automated hints for good table design
- Try ALTER, report results
- Give developers a playground that looks like production systems

Testing: Algorithm=INSTANT

```
nicolai@db [nicolai]> SHOW CREATE TABLE example\G
*************************** 1. row ***************************
       Table: example
Create Table: CREATE TABLE `example` (
  `id` int unsigned NOT NULL AUTO_INCREMENT,
  `val` varchar(200) NOT NULL,
  PRIMARY KEY (`id`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1
1 row in set (0.00 sec)

Nicolai@db [nicolai]> ALTER TABLE example MODIFY val VARCHAR(400) NOT NULL, ALGORITHM=INSTANT;
ERROR 1846 (0A000): ALGORITHM=INSTANT is not supported. Reason: Need to rebuild the table to change column type. Try ALGORITHM=COPY/INPLACE.
```

Booking.com

Avoiding ALTER

- ”Schemaless” isn’t
  - Programs = Algorithms + Data Structures
  - Schema just moves elsewhere
- The dumber the query, the slower it is
- Need a data model that is **fast and flexible**

Relational + Document

- Put required data in relational columns
  - And index it as needed
- Put optional data into JSON
- Avoid virtual columns that you index
  - Because then the indexed element has to be there for efficiency and you just removed *optional*
  - Fix your code instead

Replace ENUM with side table for equal speed

- **ENUM**

```
CREATE TABLE product (… vegetable_name ENUM ('carrot', 'turnip', 'tomato', 'onion'), …)
```

- **Side table**

```
CREATE TABLE product (… vegetable_id tinyint COMMENT see vegetable table, …)
CREATE table vegetable ( id tinyint, vegetable_name name VARCHAR(32), ...)
SELECT ... v.name ... FROM poduct p
JOIN vegetable v ON p.vegetable_id = v.id
WHERE...
```

Booking.com

Can you 10x this?

- Design the schema for growth
- Running out of space in ID columns is painful
  - Especially `AUTO_INCREMENT`
- Can you still alter a 10x size table?
  - Need space to rebuild it
  - Improve data model to avoid very large tables

Can you audit this?

- Someone **will** ask you to prove all changes are authorized
- Collect schemas and track differences
- Correlate with change tickets
  - Well-defined change format is better than free-form code diffs
- Compensating control
- Reduce audit work to exceptions only

?

nicolai.plum@booking.com

https://jobs.booking.com/careers

ORACLE DevLive | Level Up | MySQL Summit

Please rate this session.

Session ID – MS03