Hyrise-R: Scale-out and Hot-Standby through Lazy Master Replication for Enterprise Applications

David Schwalb, Jan Kossmann, Martin Faust, Stefan Klauck, Matthias Uflacker, Hasso Plattner
Hasso Plattner Institute, University of Potsdam, Germany
IMDM 2015
Motivation

New enterprise applications ..
- Growing number of users
- Increasingly complex queries
- Interactive data exploration
.. require scalability

Scale-up vs. scale-out

(+ availability)
Related Work

Theoretical replication models and comparison

- Eager vs. lazy
- Group vs. master

Implementations

- Postgres-R – Eager group replication based on shadow copies
- ScyPer – Lazy master replication with row layout for master node
- ..
**Storage engine** developed at **HPI for research** and prototyping, initially focused on main memory processing and **hybrid storage layouts** of tables

- Dictionary and bit-vector compression
- Main/delta architecture with merge process
- Hybrid row and column layouts of tables
- Supports vertical and horizontal partitioning
Dispatcher

Redirect queries to cluster nodes
- Transactional workload -> master node
- Reads -> all cluster nodes
Replication Mechanism

Logs are written to file system + send to cluster interface
Cluster interface sends *(dictionary encoded)* log information to replicas
Frequency is configurable and based on
- Number of calls
- Exceeding buffer size
- Time since last transmission

TCP with nanomsg
- Survey pattern allows replicas to acknowledge reception
- Heartbeat protocol for failover
Why Hyrise-R is a good fit for Enterprise Applications

- **Customers**
- **Sales Managers**
- **Decision Support**

OLAP, Search and Read-Only Applications on Transactional Schema

OLXP

OLTP

Operational Reporting & New Applications

Data Entry

Read-Only Replicas

Master Node

< 1 Second

**Why Hyrise-R is a good fit for Enterprise Applications**

**Hyrise-R**

Stefan Klauck

[1] Chart 9

---

Detailed text:

- Warehousing solutions with an ETL process, the applications use the same transactional schema.
- Contrary to existing data work on snapshots and evaluated on read-only replicas of transactional data, managers access run decision support queries.
- Profiles and use analytical applications such as recommendations.
- Customers can access the system to track the status of their orders.
- Analytical applications on top of the transactional data.
- Customers require strong transactional constraints.
- The master node handles transactions and OLXP workload with relaxed isolation levels.
- I propose that a powerful transactional database with strong transactional constraints is needed for applications such as stock level or available-to-promise checks that need to run in real-time. If we want to access historic data, we simply access both hot and cold data partitions. On the other hand, we can concentrate on hot data partitions, but we need to keep track of cold data partitions to be able to access them.
- The scale-out is performed by shipping the redo log of the master node to replications that replay transactions in batches.
- New applications, more users, and more complex queries can be supported by a powerful transactional database.
- The need for real-time access to data leads to the design of databases that keep data permanently resident in volatile memory and predicts that this database design will replace traditional row-based databases.
- Data volume management is often desirable to compare the current year with the last year, apply the same logic to last years data, and all other historical data.
- The possibility of predicting future trends or quickly reacting to real-time events will lead to completely new types of applications.
- Modern IT solutions create business value through analytical enterprise workloads.
- In the past 5 years, we have proven this happening in the market as all major database vendors have written new applications, which I have not even dreamed of ten years ago, have confirmed that experiences gained by rewriting existing applications and reworking traditional data models are concentrated on workload management features for the proposed database architecture.
- New changes to database architecture a viable solution. New changes to data models and hardware transactional memory are on the horizon and can be used as a fast storage medium for the database log. In addition, technologies such as non-volatile memory can easily be integrated into the commodity hardware stack, such as non-volatile memory column-based systems without any transaction-maintained volatile memory allowing to significantly decrease recovery times, introducing the new challenge of directly updating volatile memory.
- In the future, the primary persistence might be stored on non-volatile memory technologies.
- The availability of large capacities of main memory has been one of the hardware trends that make the proposed database architecture a viable solution.
- New changes to database architecture are possible today, but the use of transactional memory is currently not feasible due to limited resources.
- Although hierarchies can be modeled in the data schema, the capability to dynamically aggregate and for new applications including enterprise simulations and 'what-if' scenarios.
- The necessity to express complex hierarchies in standard SQL can be cumbersome and very expensive. Consequently, we plan to further investigate new approaches.
- Models and complex hierarchies querying complex hierarchies need efficient aggregation mechanisms, which are inherently supported by an in-memory database.
- Additionally, applications need to be redesigned to rapidly respond to new requests and only consume little main memory capacity.
- Operational reporting and new applications can use different representation, the replication for typical enterprise workloads can be performed with a delay of less than a second. In turn, the transactional workload is managed by the master node and the share of the read-only workload that has relaxed transactional constraints.
- Despite the aggregate cache, the increasing demand of data entry is often desirable to compare the current year with the last year.

[Image 601x327 to 709x389]
5 machines with
- Intel Xeon E5-2666 v3 (36cCPUs; 10 cores @ 2.9GHz)
- 60 GiB main memory
Conclusion

Hyrise-R – a system to cluster Hyrise instances using lazy master replication

- Dictionary compressed logs for updating replicas
- Heartbeat protocol for failover
- Benchmarks on Amazon EC2 cluster

Future Work

- Extend query dispatching and distribution
- Extend mixed workload measurements (ch-beCHmark)
References


Thanks

Stefan Klauck
stefan.klauck@hpi.de
http://epic.hpi.de