Supercharging your Legacy Systems with Kx

Hugh Hyndman, Director, IoT Solutions

Contents

Introduction

If your company is not already drowning in data now, it will be soon. Gartner Inc. forecasts that 8.4 billion connected things will be in use worldwide in 2017, up 31 percent from 2016, and will reach 20.4 billion by 2020. In 2017, six million new things will be connected every day. Each one of them will transmit a stream of data, adding up to sensor data volumes that will dwarf today's volumes. How your business uses this streaming data to compete in the market may determine its long-term success within the world of Industrial IoT.

These volumes impose significant challenges and pressures on solutions based on traditional relational database management systems (RDBMS) to ingest, process, and store this data. These systems were not built to handle such enormous volumes of data and this is resulting in significant and unsustainable increases in license fees, infrastructure, and operating costs for industrial businesses.

To tackle these challenges, organizations are turning to a high-performance database technology from Kx Systems which has been used by leading organizations to achieve low-latency and high-volume event processing. Kx's combination of in-memory and on-disk database, its efficient columnar design, and highly-integrated query language adds up to the fastest analytics engine on the market.

In this paper, we discuss approaches and techniques for preserving your investment in your software and RDBMS technology while addressing the performance, scale, and cost challenges when working with high volumes of data. We describe three architecture patterns (upstream, midstream, and downstream) for implementing Kx alongside existing systems and processes, discuss their advantages and disadvantages, and provide case studies of these patterns in action.

Kx Technologies

Kx is a suite of enterprise-level products centered around kdb+, the world's fastest time series database. Kdb+ is optimized for ingesting, analyzing, and storing massive amounts of structured data. The combination of the columnar design of kdb+ and its in-memory capabilities means it offers greater speed and efficiency than typical relational databases. Its native support for time-series operations vastly improves both the speed and performance of queries, aggregation, and analysis of structured data.

One feature that really sets Kx apart is its ability to combine streaming, in-memory and historical data in one simple and unified platform; there is no requirement to acquire disparate components to build a hybrid solution.

For many of today's businesses, the promise of big data analytics is the ability to use both streaming and the vast amounts of historical data effectively. This includes data that will accumulate over future years, as well as data a business may already have warehoused but have never been able to use. The means by which a database uses data storage, both in-memory and on-disk, can make a tremendous difference in the speed and cost of analytics it can produce.

The diagram below illustrates the Kx suite of technology, all built on the core kdb+ engine. Included are standard programing language interfaces, SOA interoperability, complex event processors, and an adapter framework used for ingesting data from external data sources.

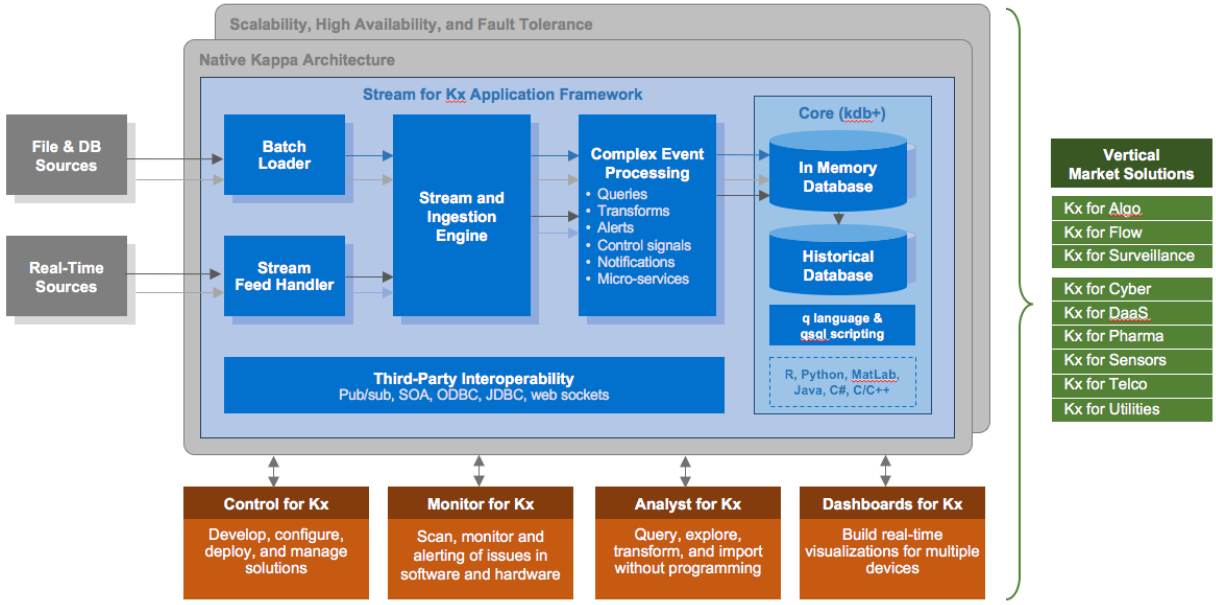

Figure 1. Kx Solution Architecture

At the end of this paper, we present some of the salient features of Kx, highlight some interesting real-world performance numbers, and provide technical information that explains why Kx vastly outperforms other solutions.

Architectural Patterns

Kx offers three architectural approaches on how its technology can be seamlessly integrated into existing systems. They are briefly summarized below, and described in full detail later in the document.

Downstream Pattern. In this pattern, data is replicated from your existing system in real time or near-time to Kx, enabling Kx to immediately take over all querying, analytic, data retrieval, and reporting functions. This option enables your existing system to remain in place to ingest and process data, and extend its useful life by off-loading data intensive functions to Kx. This pattern is also known as an operational data mart or warehouse.

Midstream Pattern. In this pattern, Kx is leveraged to process captured data and provide the results of this processing to the legacy system. This pattern places Kx technology between systems like, say, a data acquisition subsystem and its downstream consumers. The value of this is that Kx can run algorithms or complex event processing on incoming data and augment the data stream before it reaches the end applications. This approach is common for data validation, estimation, correction, and exception handling processes, with high quality data provided to the legacy system application for further processing.

Upstream Pattern. In this pattern, Kx is used as the book of record or repository for storing time-series and related data for business operations and analytics. Integration of Kx is achieved by making software changes to an application's Data Access Layer.

Downstream Pattern

The downstream pattern is a commonly used where Kx is required to act as a real-time or near real-time data mart or warehouse in order to satisfy high a demand for data queries and analyses.

In this pattern, the existing system continues to be used as the system of record. However, the time series data, as well as master or dimensional data, is replicated downstream to Kx in real or near real-time. Data is replicated from the existing database system using a change data capture process or a message bus with minimal impact to the source system. Any existing query and analysis workload can be immediately migrated from the legacy relational system to Kx and any new queries and analysis are implemented on to kdb+.

The generic Downstream Service pattern is illustrated below:

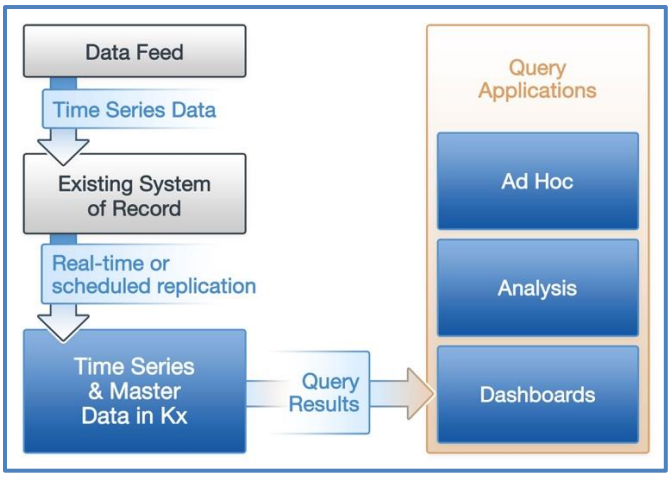

Figure 2. Downstream Pattern

Pros and Cons

With the downstream pattern, existing systems need very little modification, preserving the functionality, availability, and performance of those systems. This significantly reduces the risk and cost of implementation a solution. As analytic, query, and other workloads are transitioned to Kx, the performance of the legacy systems can be significantly improved.

Replicating data to Kx and using it for new and existing query applications is a rapid process, as Kx can ingest and transform data at many millions of records per second. Most of the integration effort is around understanding the source data, its quality, and transforming it into an appropriate data model for analytics and reporting.

Although data is replicated in a separate data store with this pattern, by shifting more workload to Kx and maintaining an on-line archive of historical data, data can be archived and removed from the source system, reducing overall infrastructure, license and operating costs for the source system. In addition, Kx provides features to make optimal use of storage. To read more about how Kx can reduce storage costs, read our whitepaper entitled “Sensor Data Storage for Industrial IoT”.

Finally, one must bear in mind that all updates must continue to flow through the existing system of record in order to minimize replication complexity and to preserve data integrity.

Downstream Pattern Case Study

The Independent Electricity System Operator (IESO) works at the heart of Ontario, Canada’s power system – ensuring there is enough power to meet the province’s energy needs in real time while also planning and securing energy for the future. The IESO, as the Smart Metering Entity, operates the provincial smart meter data processing service (the MDM/R) for 67 distribution system operators and over 4 million customers. This service processes over 100 million meter reads per day for billing customers on time-of-use rates.

The IESO was faced with a large database of smart meter data (over 250 billion meter reads and growing at over 100 million per day), a growing volume and variety of data retrieval and analytics requests, and the need for providing processed data to utilities in a timely manner. The existing transaction system and its relational database system were not originally designed to handle these increased requirements which therefore needed to be offloaded so as not to impact the core operation of the MDM/R.

After conducting market research, evaluations and proof of concept testing, the IESO found that Kx and its q programming language, delivered very high levels of performance at the lowest total cost of ownership. The IESO’s testing of technologies involved simulated data representing 260 billion meter reads for 5 million meters and over 5 years. IESO conducted over 25 tests on this data including data ingestion, on-demand retrievals, aggregations and research analytics. Across them all Kx came out on top.

Implementation Architecture

The IESO implemented a Data Mart using Kx technology to support the current volumes of data access requests and anticipated growth, 24x7x365 availability, and current and future demands for advanced analytics in Ontario's energy sector. The diagram below depicts a high-level schematic of how Kx integrated in their environment.

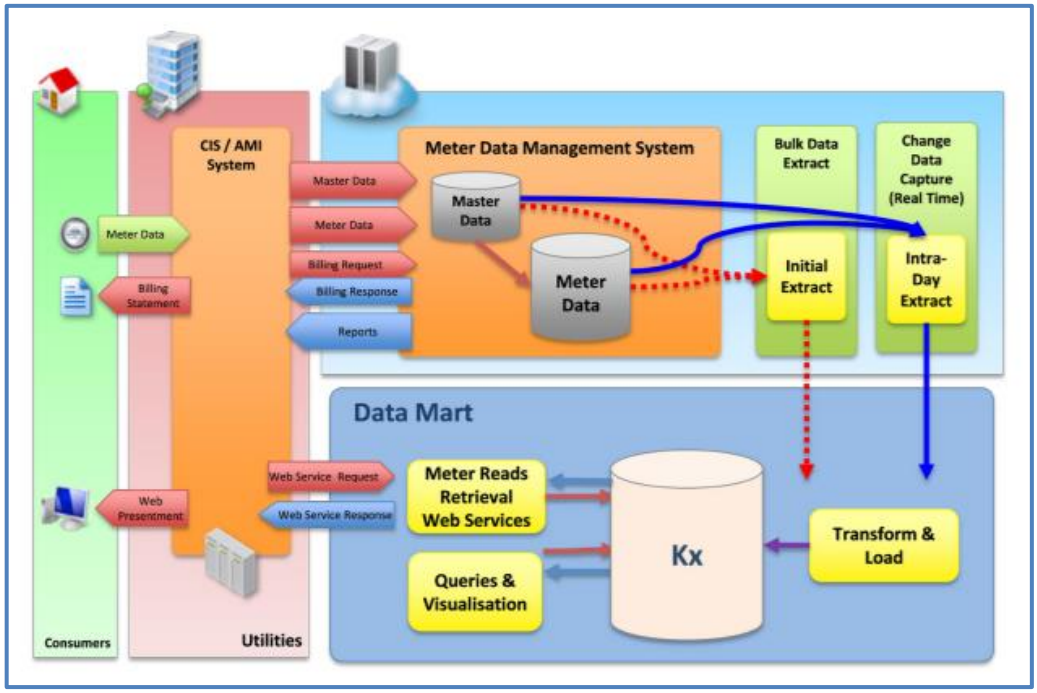

Figure 3. IESO Data Mart

In order to get data out from its existing source database system into a Kx-based data mart, the IESO deployed a change data capture tool to extract data from the source database system and deliver this data as flat files.

As soon as the data is available from the source database, Kx ingests the data, performs a number of validation checks, inserts it into a new data model, and makes it immediately available for data retrieval and analytics.

Midstream Pattern

In this pattern, Kx is implemented as a midstream service to analyze and enrich a data stream for down stream systems. It pre-processes data received from data capture systems as a stream or in batch, and then feeds the result of this processing to existing transaction processing systems.

As part of this enrichment process, Kx validates and transforms data based on business rules, adds additional information and fields, identifies and takes action based on exceptions, and notifies other systems and operators of exception conditions.

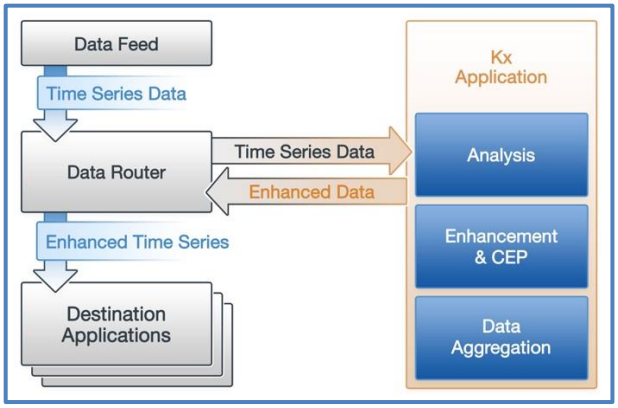

Figure 4. Midstream Pattern

Pro and Cons

Similar to the downstream pattern, the existing technology is maintained. There is minimal impact or changes required to existing applications as Kx is inserted into the transaction processing stream for pre-processing and enhancing data for use by other systems. A subset of the data will need to be replicated in order to process the incoming data stream some additional storage and licensing costs will be incurred.

This pattern can be implemented quite rapidly. The level of effort to route data to Kx and start enriching the data stream is relatively low.

However, as the existing system continues to bear the brunt of recording high data volumes. If its performance is marginal, this architecture will provide little or no relief.

Midstream Pattern Case Study

A consortium of financial institutions formed an association to process debit card payments. The association runs a large network that connects ATMs, POS devices and banking systems. The network strives to maintain

high availability and low latency. Very little processing, other than routing the transactions, is currently done on the network. The association wanted to expand its value-added services and offer its members a centralized fraud processing capability. The design for the fraud system involved a midstream architectural pattern.

Implementation Architecture

When implementing the midstream architecture, the financial consortium made clear that the sensitive information on each debit card was not to be stored in the fraud detection system and that this system itself was to be designed as a standalone service with no ability to access the databases of the member organizations. In addition, the system needed to return a thumbs up/thumbs down recommendation whilst keeping network latency at a minimum

As a result of these privacy and speed requirements, the association designed a fraud detection system using this pattern. Middleware rerouted relevant message to the system for analysis of recent purchase patterns associated with the debit card, and passed on a “thumbs-up/thumbs-down” recommendation to the subscribing member financial institution.

There were several key aspects to the system:

- The debit card number, date/time stamp, location of the transaction and amount were the only data elements stored in the database.
- The fraud detection service was assigned a strict time budget for its analysis. The system was engineered to return a recommendation 95% of the time within the budget. If the time budget was exceeded, a default “thumbs-up” was returned.
- The detection of fraud was behavior-based and required no data from the member institutions.

Upstream Pattern

This case study illustrates a common pattern where Kx is implemented as an upstream data processing and analytics service to directly address performance and licensing cost challenges with conventional RDBMS-based systems. The approach involves Kx being introduced as the system of record for time series data whilst reducing the role of the existing RDBMS to continue to be the primary repository of master data. Where possible, the application layer will be isolated from the data layer by using a DAL (Data Access Layer).

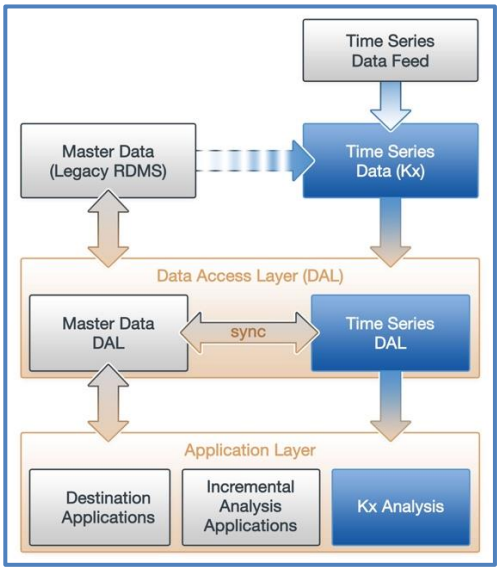

Figure 5. Upstream Pattern

Pros and Cons

This pattern provides a significant number of advantages, as summarized below.

- *Improved Performance* - The bottleneck in processing a high volume of time series data is eliminated by using Kx in this role.
- *Reduced Costs* - The RDMS licensing can be significantly reduced, offsetting any development costs.
- *Investment in existing technology* - This is maintained for the storage of master data.
- *Rapid implementation* - If a DAL exists already, the level of effort to convert the gathering of time series data to Kx is fairly low.
- *Lower risk* - If a DAL exists, the level of risk in converting the handling of time series data to Kx is low.
- *Accelerated Development* - Once Kx is in place, additional functionality can be developed cheaply and rapidly using the powerful q language.

- *No Data Bloat* - There is little or no replication of data involved in this pattern

The disadvantage of this approach occurs when the legacy software has been designed without a DAL in place and, as a result of this, may require extensive modifications.

Upstream Pattern Case Study

A manufacturing software company was having difficulties with its underlying RDBMS technology; its software was unable to cope with the rapid growth of data originating from equipment on the factory floor. The increasing volume and velocity of data introduced significant latency issues during ingestion and complex event processing, and also prevented users from getting timely analytics.

The company attempted to fix the performance problems by specifying faster servers, exploiting RDBMS enterprise features, making use of more processor cores, and making constant changes to their software. However, the costs of hardware, licenses, and their own internal costs resulted in an increase in the software product's total cost of ownership – thus making their product uncompetitive from a pricing and performance point of view.

A solution was required to solve this problem that didn't involve rewriting the product. This is where the Kx upstream pattern was applied to provide an efficient way to enhance performance without making significant changes to the software platform.

Implementation Architecture

The current system was architected so that the RDBMS stored both the sensor and master data. A Data Access Layer (DAL) was used to provide an abstraction between the database and the business logic. In order to introduce Kx into the system, the architects chose to make small changes to the DAL to support the high-volume sensor data with Kx, while keeping the master data intact in the RDBMS. It turns out that sensor data contributed to 95% of the storage and processing infrastructure, but only 5% of the software logic in the DAL.

By modifying only the DAL, the company was able to save on its investment in their software, but more important, they were able to easily scale their software product to support significantly higher sensor volumes, but also reduce the cost of infrastructure and licensing required to run their RDBMS and their system as a whole. Furthermore, they were able to downgrade their RDBMS to a less expensive version since they did not need enterprise features required for large volumes of data.

The diagram below highlights the before-and-after flow of data once Kx is introduced.

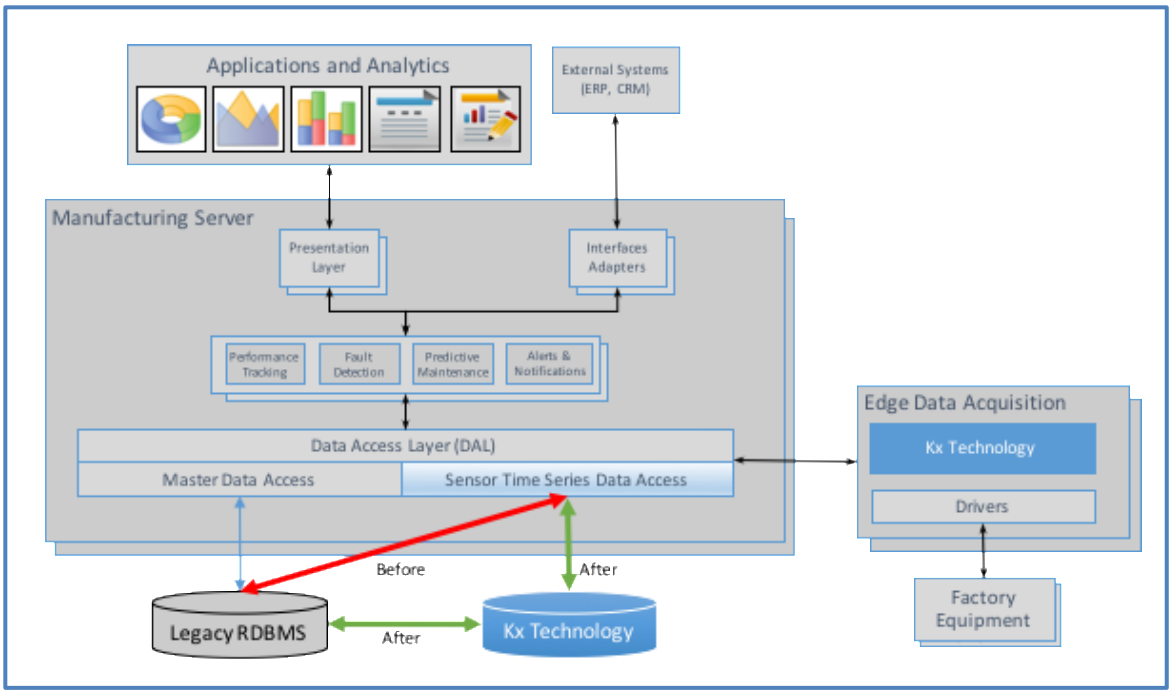

Figure 6. Manufacturing Software

What are the Key Features of Kx Technology

- The world's fastest time-series column-store database
- Streaming, real-time and historical data in one platform
- Kx runs on Linux, Windows, Solaris, and MacOS
- Kx runs on commodity hardware, cloud, edge devices/appliances
- Expressive query (qsql) and programming language (q)
- In-memory compute engine for Complex Event Processing
- Column-level compression and sensor data noise filtering
- Integrates easily into legacy systems for performance augmentation
- Multi-core / Multi-processor / Multi-thread / Multi-server

Why is Kx so Fast?

Performance is key to the underlying design of Kx, and the database has a number of important characteristics that contribute to its speed:

- Automatically distributes database operations across CPU cores
- Exploits vector instructions from Intel and ARM chipsets
- Small memory footprint (500KB) exploits L1/2 CPU caches
- Page faults through memory-mapped files
- Natively supports array operations and parallel computations
- Database tables are first class objects in q programming language
- Wide variety of data types for compact storage
- Supports software and hardware compression

Kx Performance Snippets

- **Streaming** – Process and analyse 4.5 million bulk events per second per core
- **Scan** – Search in-memory tables at 4 billion records per second per core
- **Batch** – Bulk ingest data at 5 million records per second per core
- **Store** – Accumulate 10 trillion data points (3 PB) of NYSE data
- **Usage** – Trusted by 17 of the world's top 20 investment banks
- **Volume** – Daily volumes of 1.6 TB of streaming data per day
- **Scale** – From Raspberry Pi, edge devices to 20,000 cores on AWS Cloud
- **Performance** – Top performing time-series database according to STAC Research
- **Footprint** – Tiny 500 KB memory profile (L1/2 Cache)
- **Latency** – Sub-millisecond latency for streaming event processing

Concluding Remarks

Long-term success in the realm of Industrial IoT will be dependent on a company's ability to handle the ever-snowballing amounts of sensor data being created. As has been the experience of many firms, traditional relational database systems are proving inadequate in handling this enormous amount of data; this inadequacy is reflected in the significant increases required in license fees, infrastructure and operating costs.

Those industrial firms who prove forward-thinking enough are jumping ship before they drown in their data. By turning to the high-performance, low latency technology provided by Kx, industrial firms can make use of its combination of in-memory and on-disk database, efficient columnar design, and highly-integrated query language to supercharge their legacy system with the fastest analytics engine on the market.

As described, the versatility of Kx allows for a number of architectural approaches that helps preserve the existing investment in your current technology stack, whilst solving the performance, cost and scaling issues associated with higher data volumes