WHITE PAPER

INTELLIGENT BUSINESS STRATEGIES

IoT Analytics Architecture

By Mike Ferguson
Intelligent Business Strategies
February 2018

Prepared for: TERADATA.

EB-7306

Table of Contents

An Introduction to IoT

Digital transformation is dominating IT investment

In almost every vertical industry today, companies are making huge investments in digital transformation to change how they operate and to create new insights to cut costs, retain customers and grow revenue. The programme of transition to a digital enterprise covers a range of initiatives which at a high level include:

- Digital transformation of operational systems and processes.
- Digital transformation of analytical systems.
- Closing the loop between analytics and operations.
- Digitisation of content.

Telemetry data is being collected to provide insights into business operations

Part of the digital transformation of operational systems and processes is the introduction of telemetry to collect new data to monitor and provide new insights into business operations. For example, to monitor and manage production lines, supply chains and the utilisation of assets to maximise benefit.

The Internet of Things is here and growing with billions of devices set to come on-line in the next few years

In addition, we are now in the era of the Internet of Things (IoT) where 'smart' products have the ability to collect and emit data that can be analysed in real time and offline to provide insights into product performance and utilisation behaviour. Evidence of this is reflected in the ever increasing number of 'things' connecting to the Internet. Sources estimate that around 8.4 billion sensor enabled devices are already in use in today[1] with over 20 billion connected devices forecast by 2020[2].

Both industrial and consumer IoT is flourishing

Generally speaking, this fast growing IoT market can be broken up into two broad categories -- industrial IoT and consumer IoT. The former includes heavy machinery, factory production lines, transportation, energy and smart cities. The latter includes wearables, phones, televisions and appliances that enable assisted living, home monitoring and home automation.

Industrial IoT is growing in manufacturing, oil and gas, logistics, retail and utilities

Industrial IoT enables continuous business optimisation

Consumer IoT enables monitoring of health, location, home energy and product usage

In industry, deployed sensors emit data to enable us to measure everything from temperature, light, vibration, movement, pressure, chemicals, airflow, liquid flow, location (e.g. GPS in smart phones, vehicles) and more with manufacturing, oil and gas, logistics, retail and utilities heavy investors. In manufacturing, analysing industrial sensor data can help companies create 'digital twins' (an exact digital representation of a physical system) where real-time simulation and modelling of a system can prevent equipment from failing, predict problems and optimise performance on a continuous basis. This means companies can optimise operations and reduce risk. On the consumer side, sensor data can help monitor health and fitness, location, home energy consumption and product usage -- all of which help deepen understanding of consumers while also helping product managers understand how to improve product design and development.

IoT brings new challenges and requires architecture extensions

The promise of IoT is clear. But to utilise IoT effectively to maximise business impact means overcoming new challenges, meeting new requirements and extending existing architecture to accommodate IoT data collection, preparation and analysis. This paper addresses these issues and looks at what Teradata is offering to enable success.

[1] http://www.gartner.com/newsroom/id/3598917
[2] www.i-scoop.eu/internet-of-things

THE CHALLENGE OF IOT

As with anything new in technology, the Internet of Things brings new challenges that many of us have not seen before nor have experience of how to overcome. These include challenges associated with devices, IoT data and where to best process and analyse that data. Let's look at these in more detail.

DEVICES

Automatic discovery of new devices and the data they generate is the only way to keep pace with new devices coming online

Many different types of sensor devices are being deployed in business operations and embedded in smart products being launched on the market. While data from these devices offers great opportunity, the challenge is keeping pace with the number of devices being deployed together with the schema describing the data they capture and emit. Given the sheer number of devices being introduced, automatic discovery of new devices and the data they emit is needed - this is the only way to keep pace with change.

IOT DATA CHARACTERISTICS

IoT data is in-motion time series data streaming into the enterprise

IoT data can be high volume and high velocity

Edge gateways collect IoT data and send it out in JSON format

Gateway emitted JSON data can change schema without notice

New devices can introduce new message content without notice

Data can be continuous or arrive in bursts

There are different types of time series data

Data can be out of sequence or missing

With respect to IoT data, this is not like just introducing another data source to an ETL tool. IoT data is data in-motion, streaming into analytic applications and computing environments from hundreds of thousands to millions of endpoints. It has unique characteristics that offer up a number of new challenges for most organisations introducing IoT processing for the first time. At a base level in an IoT environment, each sensor emits time series data that includes a sensor ID, a timestamp and at least one measure. What's unique about this is that:

- It is captured and emitted in a wide range of formats with no standards. Therefore, a new type of sensor can introduce a totally new format.
- Data is typically 'in-motion' and streaming at very high velocity.
- A lot of devices can quickly generate huge volumes of streaming data.
- Timestamps emitted by IoT device/sensors are almost always out of sync.
- Gateways are therefore often used to collect sensor readings, group them together and assign a single timestamp before sending out a single JSON message containing multiple readings.
- The number of device readings in a single timestamped message can vary and so the contents of gateway emitted JSON messages can vary without notice.
- Upgrading devices can cause message content to change without notice.
- Adding new devices can introduce new message content without notice.
- Data can arrive continuously (never stops), in heavy bursts or be captured in batch – both may be required depending on business need.
- There are different classes of time series data to contend with including:
    - 7x24 infinite (continuous) time series data.
    - Time series data with an overlaid logical interpretation e.g. a work shift, a flight, a car journey.
    - Fixed size time series e.g. a few thousand entries associated with a scientific trace of some kind.
- Events emitted by sensors can arrive out of sequence.
- Data can be missing.

Processing at the Edge Versus Processing at the Centre

IoT data can be analysed at the edge or collected and sent to an enterprise facility for analysis

In addition, with so many sensors and connected IoT devices, scalability is needed to process and analyse this data. This has given rise to the use of central scalable in-memory-based stream processing platforms to ingest, prepare and analyse IoT data. However, if you think about sensors in cars and all the cars being driven in the world or remote patient monitoring and all the people with wearable devices, should you bring all IoT data to the centre for processing? Or does it make more sense to process and analyse that data "at the edge" close to where data is created? Or should it be both? Figure 1 shows both, which is the most likely option. Edge analytics enables decentralisation of some decision making. Note also that data can be analysed in-flight before it is stored anywhere (Figure 2). But if this is the case, how do you support both? How do you integrate two types of IoT analytical processing into your existing analytical environment? How do you manage edge analytics as well as enterprise analytics? There are clearly a lot of new challenges here.

Edge analytics often happens on a gateway

Analysis of IoT data at the enterprise level can happen on-premises or in the cloud

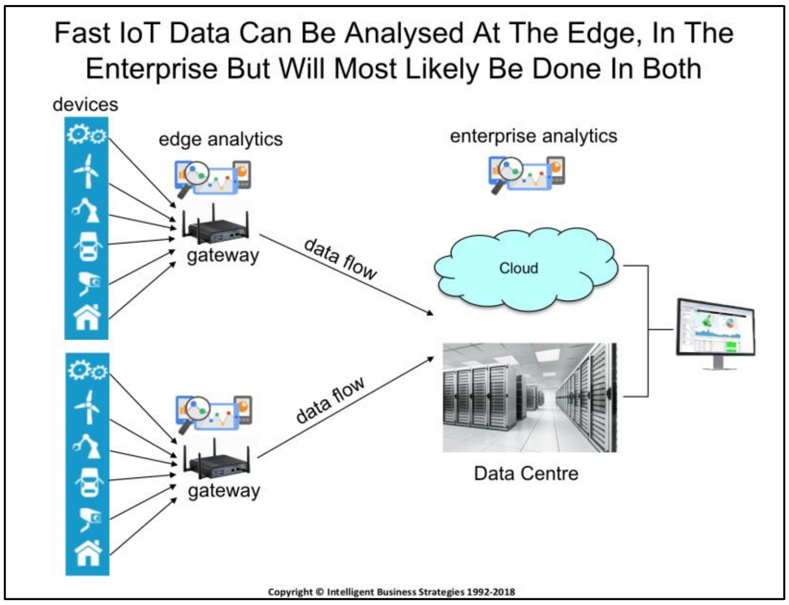

Fast IoT Data Can Be Analysed At The Edge, In The Enterprise But Will Most Likely Be Done In Both
devices
edge analytics
enterprise analytics
gateway
data flow
Cloud
data flow
gateway
Data Centre
Copyright © Intelligent Business Strategies 1992-2018

Figure 1

Streaming analytical applications typically filter, clean and integrate IoT data before feeding it into a model for automatic analysis

Automated actions may be triggered if business conditions are detected e.g. alerts, recommendations etc.

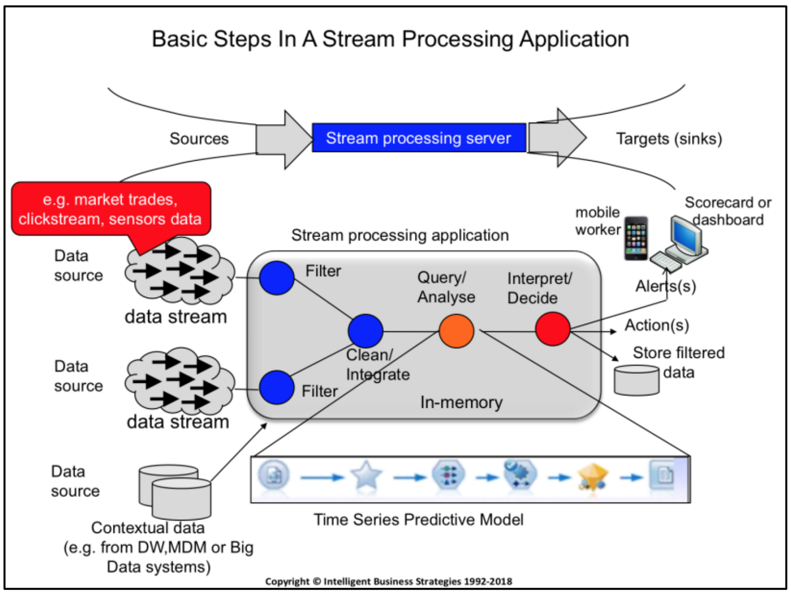

Basic Steps In A Stream Processing Application
Sources
Stream processing server
Targets (sinks)
e.g. market trades, clickstream, sensors data
Data source
data stream
Data source
data stream
Data source
Contextual data (e.g. from DW,MDM or Big Data systems)
Stream processing application
Filter
Query/ Analyse
Interpret/ Decide
Clean/ Integrate
Filter
In-memory
mobile worker
Scorecard or dashboard
Alerts(s)
Action(s)
Store filtered data
Time Series Predictive Model
Copyright © Intelligent Business Strategies 1992-2018

Figure 2

IoT Analytical Processing Requirements

Given the new challenges of processing and analysing IoT data, there are a number of requirements that need to be met in order to introduce IoT analytics into the enterprise. Some of the key ones are listed in this section.

Continuing on from Figure 1, a key requirement is that both edge and central (data centre or cloud based) IoT analytical processing are needed.

Requirements for IoT Analytical Processing at the Edge

With respect to IoT analytical processing at the edge, it should be possible to:

Automatic discovery is key in edge computing

Data needs to be transformed at the edge

Analytical models need to be deployed at the edge

- Automatically discover, classify, register and add properties about edge devices or gateways that you want to collect data from and analyse.
- Define and/or automatically discover, catalog and manage schema for data emitted by each type of device.
- Automatically detect change – new devices, devices offline, schema drift.
- Remove any private information on edge data before processing.
- Pre-process and transform raw data at the edge.
- Integrate raw data at the edge e.g. from multiple devices to see across multiple sensors. For example, Apache® MiNiFi.
- Manage master data at the edge needed to apply context during analysis.
- Deploy and manage analytical models (including version control) to execute at the edge.

Also edge rules need to be defined to manage local decisions and actions

- Define edge rules for execution at the edge.
- Manage edge rules including support for version control and parameters.
- Activate and de-activate edge rules.
- Define business conditions indicated by patterns in the data.
- Define the type of action that you want to take at the edge if a business condition is met.

Version management of models deployed in the network is critical

- Define conditions (rule combination) upon which actions should be taken.
- Drive automated actions as a result of analysing data locally.
- Track actions to provide an audit trail for governance and to provide lineage to understand what action(s) were taken and why.
- Identify conditions to send to the enterprise data centre or cloud for further analysis.

Edge analytics need to be automatically monitored and replaced if the become stale

- Filter and reduce the data sent to the enterprise data centre or cloud.
- Flow data to the enterprise data centre or cloud.
- Monitor the accuracy of analytical models deployed at the edge and trigger automatic re-training, re-test and re-deployment should accuracy drop below user defined thresholds.

Looking at these requirements it should be possible to develop data processing jobs and analytics centrally but deploy them to run at the edge.

Requirements for IoT Analytical Processing at the Centre

With respect to IoT analytical processing at the centre, it should be possible to:

Data Ingestion

Scalable data ingestion is needed

- Ingest high velocity data at scale from potentially hundreds of thousands or even millions of devices.

- Scale to accommodate bursts in data during ingestion.
- Automatically discover, profile, semantically tag, catalog and classify IoT data in a data catalog to understand meaning and how to govern it.

Data Preparation

Scalable data preparation is needed to process high volume, high velocity data

Need to deal with multiple formats, gaps in the time series and data that arrives out of sequence

- Automatically detect change e.g. schema drift.
- Remove any private information on edge data before processing.
- Prepare data at scale (to handle data volume and velocity), with the ability to deal with:
  - Multiple formats from different devices.
  - Time series data from different sensors/devices that arrive with timestamps out of sync.
  - Time series data that arrives out of sequence.
  - Data quality issues e.g. clean-up.
  - Any gaps in time series data e.g. fill the gaps.
  - Varying time series data message content emitted by edge gateways or devices.

Performant Data Access at Scale

- Partition and access ingested and prepared time series data in parallel preferably using a widely used API, such as SQL.
- Pick out data attributes from schema variant JSON based IoT data.

Logical Time Period Definition

Need to analyse data in the context of 'business time' e.g. a 6-hour work shift

- Divide up the data into one or more logical time periods (e.g. multiple work shifts to compare what happens from shift to shift, a flight, a weekday, a weekend, or a 'rush hour') to set boundaries within which some types of time series analysis will take place. This could represent a specific state for example (i.e., a time period when something is in a specific condition.).

Noise Cancellation

Need to remove data not relevant to the analysis being performed

- Perform dimensionality reduction on the data at scale to eliminate rows and columns that are not relevant to the analysis being conducted.
- Support a number of data reduction techniques at scale.

Data Integration

Data integration is needed to provide contextual data and to enable multi-variate time series analysis

- Integrate data from multiple devices/sensors to create a dataset needed to identify patterns representing one or more specific business conditions.
- Manage master data needed to apply context during analysis.
- Integrate IoT data with master data such as asset/equipment, product, or customer.
- Integrate IoT data with geospatial data (e.g. for mobile assets.)
- Integrate IoT data with other activity data (e.g. combining the monitoring of oil and gas flows with drilling data) to provide context for the time series analysis.

Feature Engineering

- Provide the ability to create new data fields from data attributes at scale that could help improve analysis of IoT data and drive new entries in a data dictionary and data catalog (e.g. from a date field you could create the year, month, week, day of the year, day of month, or day of the week.)

Data Storage

Need to store semi-structured data in an optimal way for analysis

- Store IoT data in a form/ forms optimised for both univariate (look back along time series data from a single device) and multi-variate (looking across time series data from multiple devices) time series analysis.
- Be able to accommodate schema variant data in JSON or other semi-structured formats e.g. AVRO, XML, or others.
- Be able to accommodate IoT data in CSV files.

Analysis and Visualisation

Both near real-time and offline analysis are needed

- Perform time series analysis of IoT data in near real-time and also offline.
- Look along a time series back through time performing univariate time series analysis at scale.
- Maintain a time series history to facilitate looking back through time to find similar events.
- Look across time series data ingested from multiple devices to perform multi-variate time series analysis at scale.
- Join IoT data to other data even if data is persisted in multiple data stores.

Need to analyse continuous and specific subsets of time series data at scale

- Analyse 24x365 infinite high velocity time series data at scale.
- Analyse fixed size (e.g. last hour) high velocity time series data at scale.
- Analyse time series data at scale within the boundaries of a user-defined logical view e.g. an 8-hour work shift.
- Use unsupervised machine learning models at scale on time series data to identify outliers within a time window and to cluster together similar patterns going back through time.

Both predictive and prescriptive analytics are required respond in a timely manner

- Use supervised machine learning models at scale to predict business conditions (e.g. equipment likely to fail).
- Use supervised machine learning models together with decision rules to analyse data at scale and trigger automated actions.
- Make use of temporal SQL datatypes and functions to analyse IoT data.

Replay

Replay is needed to analyse pattern recurrence over time

- Replay streaming time series data to tolerate machine failures during analysis and to go back in time looking for specific patterns in data either in batch or to compare with a pattern identified in near real-time analysis.

REQUIREMENTS FOR INTEGRATION WITH EXISTING ANALYTICAL SYSTEMS

In addition, to IoT analysis at the edge and in the data centre, IoT analytics need to integrate with existing analytical systems. To enable this to occur it should be possible to:

Need to integrate IoT analytics with your existing analytical environment

- Develop and monitor analytics centrally on data in a data warehouse or in Apache® Hadoop®/Apache Spark™ for deployment or the trained model at the edge.
- Analyse edge data from sensors and gateways in near real-time at scale.

Need to filter IoT data of interest into Hadoop or into a data warehouse for subsequent offline analysis

- Filter edge IoT data of interest into Hadoop or a data warehouse analytical database for further batch and offline analysis.
- Query across real-time IoT time series data and data in data warehouses or MDM systems.
- Build views on top of IoT data to identify state to simplify access.

Join IoT data to contextual data in data warehouses or MDM systems

- Access IoT insights, predictions and insights from self-service BI tools and from other applications.
- Publish decisions as a service in an information catalog.

AN ARCHITECTURE FOR IOT ANALYTICS

IoT data can be analysed at the edge and in the enterprise

IoT data can also be persisted into Hadoop and/or a data warehouse for scalable offline analysis and joined to other data in your existing analytical environment

With respect to architecture, Figure 3 shows an architecture that accommodates IoT data plus traditional and other big data analytics.

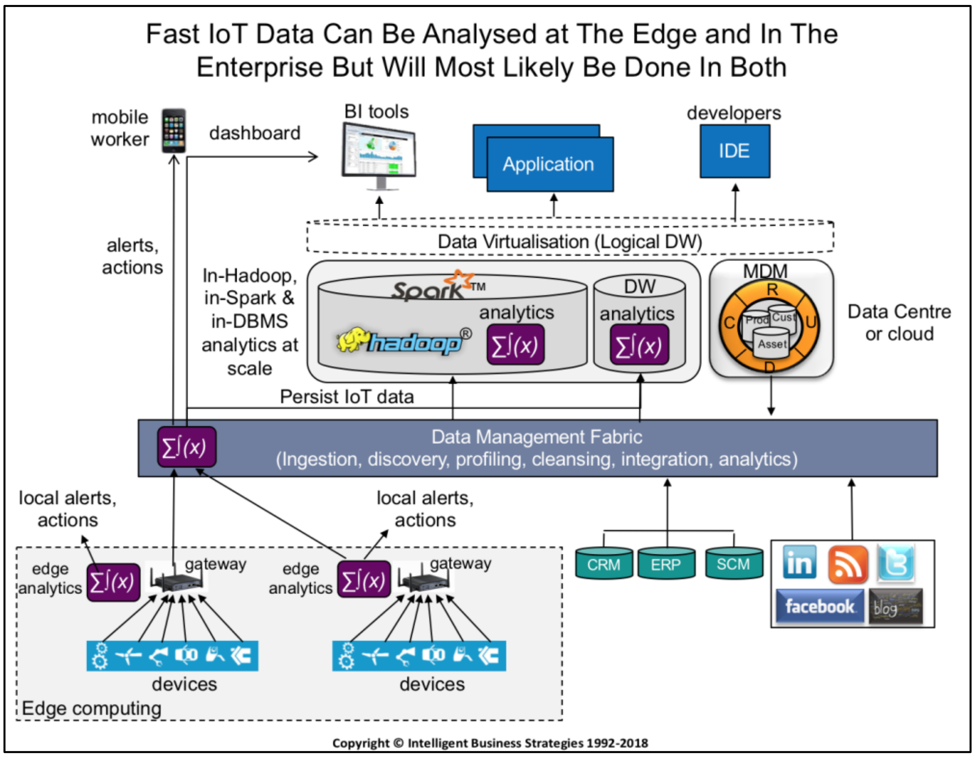

Fast IoT Data Can Be Analysed at The Edge and In The Enterprise But Will Most Likely Be Done In Both
mobile worker
dashboard
BI tools
Application
developers
IDE
alerts, actions
Data Virtualisation (Logical DW)
In-Hadoop, in-Spark & in-DBMS analytics at scale
Spark
analytics
hadoop
DW analytics
MDM
Data Centre or cloud
Persist IoT data
Data Management Fabric (Ingestion, discovery, profiling, cleansing, integration, analytics)
local alerts, actions
edge analytics
gateway
devices
Edge computing
CRM
ERP
SCM
facebook
blog
Copyright © Intelligent Business Strategies 1992-2018

Figure 3

Data in data warehouses and master data management systems can be integrated with IoT data to provide context for analysis and to help produce business insights

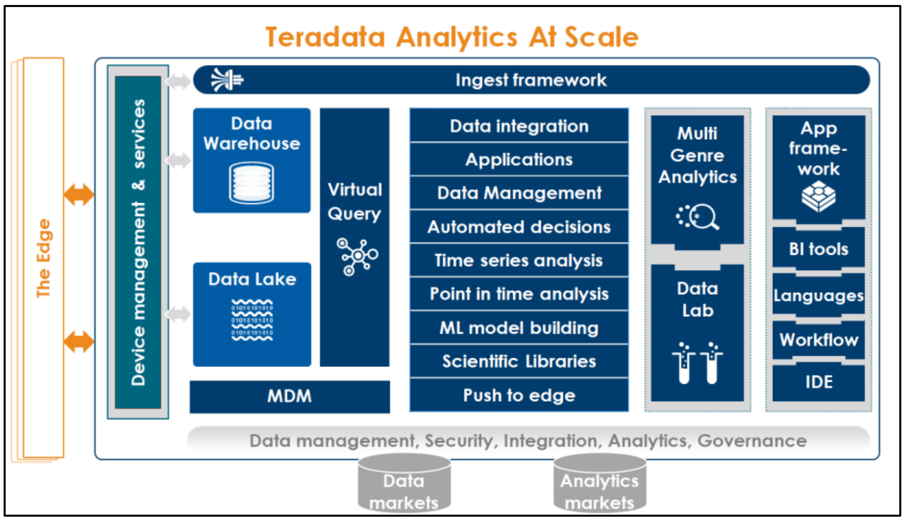
This shows that IoT data can be analysed at the edge in real-time to trigger local action(s) on a single instance of a pattern indicating a business condition. Also, IoT gateways can package up the readings from several devices into single messages and send them into the data centre or enterprise cloud computing environment. This data can then be analysed in-flight in near real-time prior to persisting the data or the data can be persisted first in a data warehouse or Hadoop and analysed at scale e.g. in Spark. Master data is shown to provide contextual data needed for the analyses. Also, the data management fabric software can integrate data pushing down jobs to execute at scale on underlying massively parallel analytical databases or Hadoop/Spark. It may also be possible to have analytical relational databases with built in support for data cleansing and integration.

IoT Solutions From Teradata

Given the unique challenges of IoT time series data and the types of analyses required, this part of the paper looks at how Teradata Corporation has extended its analytical relational database and accompanying products to offer IoT solutions.

Figure 4 shows the Teradata architecture with supports for IoT analytical processing:

Teradata's architecture accommodates analytics at the edge and scalable IoT analytics in the enterprise



Source: Teradata Figure 4

Teradata IoT analytical processing is primarily focussed at the centre, either on-premises or in the cloud. It does however support development of models that can be deployed at the edge. More on that later. For now, let's look at the centre.

Teradata IoT Analytical Processing at the Centre

Teradata's architecture accommodates analytics at the edge and scalable IoT analytics in the enterprise

It also facilitates integration of IoT data with existing analytical systems

Teradata analysis of data from things includes support for:

- Data ingestion.
- Storage of IoT data in the Teradata® Database(running on-premises or in the cloud), on cloud storage or in Hadoop
- Semi-structured data types in the Teradata Database
- In-database time series data preparation at scale
- In-database time series analysis at scale
- Joining of IoT data to non-IoT data

Data Ingestion

Teradata has two ways to ingest high velocity, high volume streaming data

Teradata provides three major ways to ingest IoT data:

- Teradata Listener™.
- Teradata Kylo™.
- Traditional batch data integration.

Fast data can be ingested from scalable messaging platforms like Kafka

Teradata Listener is a self-service solution for ingesting and distributing extremely fast-moving data streams in near real-time. It can ingest data from RESTful web services (e.g. applications) and MQTT data sources. MQTT data sources use a central broker with messages organised by topic (labelled queue). Examples here would be Apache Kafka®, Apache ActiveMQ and Eclipse Mosquitto™. That means any kind of streaming data published to these scalable message queuing technologies can be ingested using Teradata Listener whether it be clickstream, social network data, IoT sensor data, data from applications published to a message topic, or data from other data stores turned into data streams.

Teradata Listener can ingest fast data into the Teradata Database, Hadoop HDFS or HBase

With respect to write processing, Teradata Listener can write data streams to a variety of targets, including:

- Teradata Database 14.10 and later.
- Teradata Analytic Platform
- Apache HDFS™ (on Cloudera® and Hortonworks® distributions)
- Apache HBase™ (on Cloudera® and Hortonworks® distributions)

Sources can also be streamed directly to external applications such as BI tools. Listener supports the ability to queue bad records (e.g. with an invalid payload) as well as the ability to continuously monitor incoming data streams, gather metadata, and show metrics for data flow over time, such as latency, number of records, and size of record. It also provides an audit log of all activities and a REST API to programmatically access and maintain sources, users and targets as well as status information.

Teradata Kylo is self-service data ingestion software that can ingest, discover, profile, validate, standardise, clean and transform data

The second option for IoT data ingestion is Teradata Kylo. This is self-service data ingestion software that allows business users (as opposed to data engineers) to automatically ingest, discover, profile, validate, standardise, clean and transform data from databases, file systems and websites into Hadoop. - including into Apache Hive™ tables. Automated discovery of data sources is made possible through the use of elastic search. Also, authorised users can add data validation rules and also join data from multiple data sources.

Teradata Kylo generates scalable data ingestion jobs and is integrated with Apache Ambari and Cloudera Manager

Once defined, Kylo generates Apache NIFI ingestion workflows that leverage Apache Spark for scalable data processing before loading data into Hive tables. Generated NIFI jobs can also be monitored using operational metrics and metadata lineage is available to provide traceability. Kylo is integrated with Ambari and Cloudera Manager to simplify administration of generated jobs in a Hadoop environment. It also integrates with Apache Sentry™, Apache Ranger™ and LDAP for security. Once IoT data is written into Hadoop Hive tables, it can be accessed from BI tools, Teradata Database and Teradata Analytic Platform for further analysis.

The use of Teradata Listener and Teradata Kylo together allows IoT data plus master data for example, to be brought together into an analytical environment to provide context for IoT analytics.

The Role of Sensor Data in a Data Warehouse

Edge analytics on its own is not enough

Most businesses want to look back on time series data to look for repetition of business conditions occurring

Persisting IoT data in a data warehouse enables sensor data to be analysed in a business context

Looking back at Figure 3, it can be seen that IoT data can be captured and analysed in real time at the edge while the data is in motion. In addition, IoT data from multiple devices and gateways can also be analysed in near real-time prior to persisting the data in an analytical data store either on-premises or in the cloud. Figure 3 shows that the data could be persisted in Hadoop or in a data warehouse. This is to signify that analytics at the edge is not enough. Business people want to do more than act on a single instance of a business condition when a pattern in the data is detected. Replay and offline analysis is also needed to look back on time series data to find repetition of business conditions occurring and much more. A key reason for persisting IoT data in a data warehouse is to enable sensor data to be analysed in an overall business context. For example, to understand asset performance and asset failures over time. Given that sensor data is often semi-structured (e.g. JSON, AVRO data) it could be argued that analytical relational databases would not be ideal. However today, scalable analytical relational databases support semi-structured data.

Time Series Analysis In The Teradata RDBMS

The Teradata Database has been enhanced to support analysis of time series data

Given this requirement, Teradata have made significant extensions to their flagship massively parallel relational database to make it possible to analyse data generated by the Internet of Things. These include:

- Semi-structured data types.
- Time series tables which make dealing with sensor data a lot easier.
- EXECR Operator (for in-database execution of R code - e.g. for dimensionality reduction).
- In-database temporal data types and temporal SQL functions which can be used with sensor data.
- In-database analytical functions invoked via SQL.
- Teradata geospatial indexing, geospatial columns and SQL extensions which can be exploited when analysing mobile things.
- The Script Operator (for in-database execution of Python).

Let’s look at a few of these in more detail in the context of IoT analytics.

Semi-structured Data Types

Semi-structured data types like JSON, XML and AVRO allows schema variant time series data to be stored in the database

A key capability in processing and analysing IoT data is support for semi-structured data. We have already discussed that IoT gateways often package up several device readings in a single JSON message with a single timestamp. Also, the number of readings in the metrics payload of a JSON message can vary and so there is a need to support schema variant data. Teradata support for JSON, XML, CSV, Avro data types in the database means that they can store IoT semi-structured data (including schema variant JSON) directly in Teradata tables. In addition, Teradata SQL supports ‘dot’ notation to enable analytical queries to ‘step into’ nested and schema variant JSON to retrieve the data.

Time Series Tables

Teradata have created new Time Series tables to enable time series data to be analysed using advanced analytics

Another new capability in the Teradata Database is the addition of time series tables to really improve productivity and reduce time to value when preparing and analysing IoT time series data. Time series tables help to organise time series data for fast performance analysis and advanced analytics. All three different classes of time series data, outlined earlier in IoT Data Characteristics, are supported. These are:

Continuous, fixed size and logical interpretations of time series data are all supported

1. 7x24 infinite (continuous) time series data.
2. Time series data with an overlaid logical interpretation e.g. a work shift, a flight, a car journey.
3. Fixed size time series e.g. a few thousand entries associated with a scientific trace of some kind.

Continuous time series data can be divided up into time intervals for analysis

With respect to infinite or long-running time series data (e.g. buoys collecting temperature and bathymetry data on a 7x24 basis), the longer the time series is, then the more likely you are to want to break it up into intervals and look at each interval. Teradata time series tables allow you to do this.

Teradata time series tables support three types of storage distribution and two types of in-table ordering. Table 1 shows the three types of storage distribution. It is up to the customer to choose the appropriate distribution strategy to suit their needs.

Data can be partitioned in one of three ways to enable parallel execution of time series queries

Teradata Time Series Table Storage Distribution Strategies

Option	Designed For	Comments / Examples
By Time Interval Only	A single time series	e.g. sensors in a building
By Time Interval AND Column List	Multiple devices producing data	Uses an Identifier for the series and the interval of time (e.g. 1 hour)
By Column Only	Collecting data for example on behalf of a car or a plane with a trip (from A to B) overlaying it	AutoID, Trip-ID for a car or a flight (PlaneID) then all records would be on 1 AMP Good if you want to compare trips All records will be stored in time order

Table 1

The two storage strategies supported in Teradata time series tables are:

Data is stored in timestamp order

Storage Strategy	Comments
Ordered by time	Data is stored in timestamp order
Ordered by time AND sequence number	Data is stored in timestamp order and sequence number within timestamp The sequence number is for machine generated data when data is being generated at a rate faster than the hardware clock. In other words, the timestamp is not enough on its own

Table 2

Time series tables (known as Primary Time Index tables) are created using a new type of index called a Primary Time Index which is optimised for time series queries. This makes them time aware with respect to storage distribution strategies and the in-table storage strategy. The syntax is shown here:

```
CREATE  time_series_table_name>  …
    PRIMARY TIME INDEX
        ( <timecode_dt> [, <timezero_date>] [,<timebucket_duration>]  [, <columns_clause,>]);
```

Teradata assigns a number to each time interval to create numbered time buckets

There are four elements that need to be defined in a time series table.

Element	What is it?	Comments
Timecode_dt	Typically a timestamp	
Timezero_date	This is the beginning of the time series	Think of it like a stick in the ground upon which everything is based
Timebucket_duration	The interval size e.g. 1 minute, 2 hours, a 6-hour work shift, 1 day, a weekend	You break the time continuum into interval time from time zero e.g. an interval of 2 hours. Teradata then assigns each interval a number so they have numbered time buckets.
Columns List (optional)	Other columns e.g. column A and column B	A, B and the time bucket number are then hashed so that everything in that time range goes to the same AMP.

Table 3

Teradata then partitions the data across the AMPs by time buckets number to enable parallel query processing

The time buckets enable the storage distribution of the associated data row to be based upon the time bucket interval instead of by timestamp and so reduce the number of the partitions on disk, and the CPU and IO needed to access the data in each query. Time buckets are vital for timestamped sensor data. Also, time buckets only apply when you store by time interval only or by time interval and column list.

Time series tables support both univariate (single sensor) and multi-variate (multiple sensors) time series analysis

It is possible to have both univariate (single variable) time series tables and multi-variate (multiple variables) time series tables. So, for example if you wanted to create a multi-variate time series table to analyse sensor readings gathered from a car journey it might look something like this:

```
CREATE TABLE AUTOTRIPS
        (TRIPID                 INTEGER,
        RPM                     FLOAT,
        ENGINE_TEMP             FLOAT,
        OIL_PRESSURE            FLOAT)
PRIMARY TIME INDEX(TIMESTAMP(6),  DATE ’1970-01-01’,  COLUMNS(TRIPID),
NONSEQUENCED);
```

This would result in everything in the one car journey ending up on the same AMP[3]. Therefore, analysing the data from a single journey would be a single AMP operation.

When you create a time series table, Teradata Database automatically generates two virtual system columns (signified by the $ sign). These are:

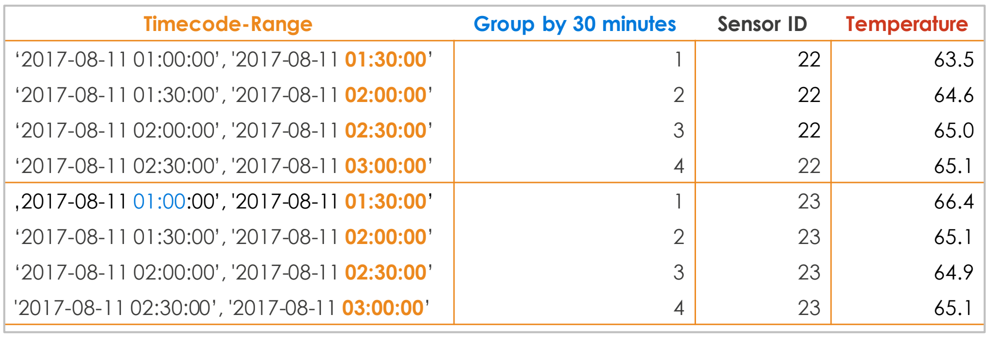
- $TD_TIMECODE_RANGE
- $TD_GROUP_BY_TIME

Time series data can be grouped by time to group data into user defined time intervals within a time range

An example of a time series query that analyses sea temperature data produced by buoys over a two-hour period and that uses these columns is shown below:

```
SELECT $TD_TIMECODE_RANGE, $TD_GROUP_BY_TIME,
          SENSORID,  AVG(TEMPERATURE)
FROM BUOYS
WHERE TIMECODE BETWEEN TIMESTAMP '2017-08-11 01:00:00'
          AND TIMESTAMP '2017-08-11 03:00:00‘
GROUP BY TIME( MINUTES(30) AND SENSORID)  USING
          TIMECODE(TD_TIMECODE)
ORDER BY SENSORID, $TD_GROUP_BY_TIME;
```

Aggregate metrics can be calculated for each sensor for each user defined time period within a time range



Timecode-Range	Group by 30 minutes	Sensor ID	Temperature
‘2017-08-11 01:00:00’, ‘2017-08-11 **01:30:00**’	1	22	63.5
‘2017-08-11 01:30:00’, ‘2017-08-11 **02:00:00**’	2	22	64.6
‘2017-08-11 02:00:00’, ‘2017-08-11 **02:30:00**’	3	22	65.0
‘2017-08-11 02:30:00’, ‘2017-08-11 **03:00:00**’	4	22	65.1
,2017-08-11 01:00:00’, ‘2017-08-11 **01:30:00**’	1	23	66.4
‘2017-08-11 01:30:00’, ‘2017-08-11 **02:00:00**’	2	23	65.1
‘2017-08-11 02:00:00’, ‘2017-08-11 **02:30:00**’	3	23	64.9
'2017-08-11 02:30:00’, ‘2017-08-11 **03:00:00**’	4	23	65.1

Source: Teradata

Here you can see the $TD_TIMECODE_RANGE representing time ranges for each 30 minute group as defined in the GROUP BY TIME clause and the $TD_GROUP_BY_TIME representing the four 30-minute groups of each specific sensor within the two-hour time range set in the WHERE clause.

Teradata time series tables yield major productivity gains in preparing data

The power of the GROUP BY TIME and TIMECODE_RANGE to organise data coupled with the data being distributed by time bucket interval and stored in time order yields major data preparation productivity improvements as well as fast query performance.

[3] AMP: Access Module Processor. An RDBMS process on a cluster server that ‘owns’ a collection of data that has been hash distributed to that AMP. Time buckets are ‘owned’ by AMPs. An AMP is a unit of parallel processing, usually 30-40 AMPs per server.

Time buckets also enable fast single AMP time series query operations

Time series tables have efficient time filtered operations because they start reading at a specified time and end at another time reading the data in time sequence.
Also, the optimiser can figure out if this is a single AMP local time series operation, a group of AMPs, or a global operation (i.e., table scan).

Both regular and irregular time series data can be supported

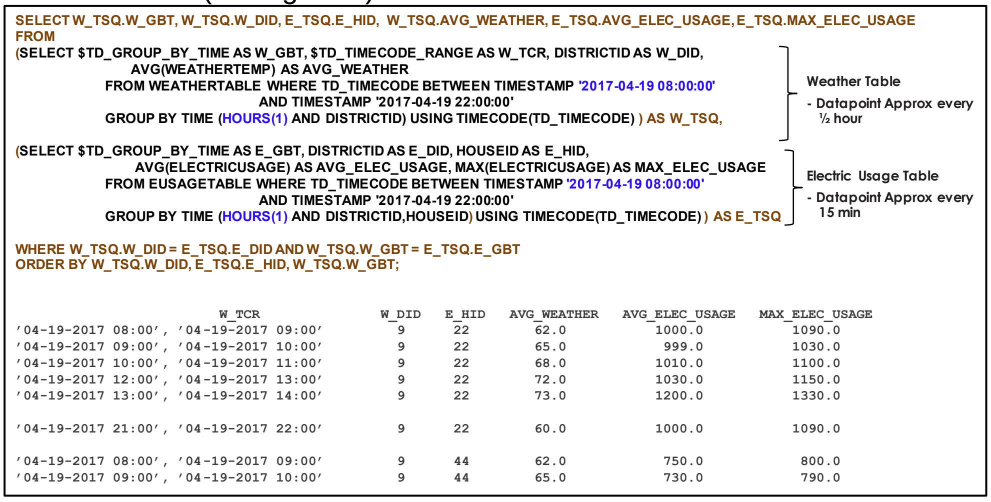
There are both regular time series (emitting of data occurs at a regular fixed time e.g. every minute) and irregular time series (emitting of data occurs randomly). There are no missing values in a regular time series. However, irregular time series are the most common, and so data can be missing. For this reason, Teradata provides a FILL option on table creation to allow a user to state what to do if the data is missing. In other words, the Teradata Database can fill in gaps in the data for you, which saves huge amounts of time for a data scientist. Looking at the AutoTrips table created earlier, an example of a time series question with the FILL clause highlighted is shown here:

There are many use cases that can benefit from Teradata time series tables and queries

Teradata can automatically fill in gaps caused by missing values in irregular time series data

```
SELECT $TD_TIMECODE_RANGE AS TCR, TRIPID AS TID, AVG(RPM) AS AVG_RPM,
AVG(ENG_TEMP) AS AVG_ENGTEMP, AVG(OIL_PRESSURE) AS AVG_OILP
FROM AUTOTRIPS
GROUP BY TIME (MILISECONDS(10) AND TRIPID ) FILL(PREV)
WHERE TD_TIMECODE BETWEEN TIMESTAMP ‘2017-05-08 08:00:00.000000’ AND
          TIMESTAMP ‘2017-05-08 22:00:00.000000
HAVING (AVG_ENGTEMP > 200.0  AND  ((AVG_OILP  < 0.8 *( AVG_RPM * 0.01))
OR  (AVG_OILP > 1.2 *( AVG_RPM * 0.01));
```

There are so many use cases for time series analysis that would benefit from Teradata time series queries including: outlier detection, stock trades monitoring and analysis, and service log analysis. It is even possible to join summary data from different time series tables where data is collected at different time intervals. For example, you could analyse the impact of weather on electricity usage by joining the aggregates from a weather time series table holding weather temperatures collected every half hour in specific regions with an electricity usage time series table holding smart meter data collected every fifteen minutes.(see figure 5)



```
SELECT W_TSQ.W_GBT, W_TSQ.W_DID, E_TSQ.E_HID,  W_TSQ.AVG_WEATHER, E_TSQ.AVG_ELEC_USAGE, E_TSQ.MAX_ELEC_USAGE
FROM
(SELECT $TD_GROUP_BY_TIME AS W_GBT, $TD_TIMECODE_RANGE AS W_TCR, DISTRICTID AS W_DID,
          AVG(WEATHERTEMP) AS AVG_WEATHER
      FROM WEATHERTABLE  WHERE TD_TIMECODE BETWEEN TIMESTAMP '2017-04-19 08:00:00'
                  AND TIMESTAMP '2017-04-19 22:00:00'
      GROUP BY TIME (HOURS(1) AND DISTRICTID) USING TIMECODE(TD_TIMECODE) ) AS W_TSQ,

(SELECT $TD_GROUP_BY_TIME AS E_GBT, DISTRICTID AS E_DID, HOUSEID AS E_HID,
          AVG(ELECTRICUSAGE) AS AVG_ELEC_USAGE, MAX(ELECTRICUSAGE) AS MAX_ELEC_USAGE
      FROM EUSAGETABLE WHERE TD_TIMECODE BETWEEN TIMESTAMP '2017-04-19 08:00:00'
                  AND TIMESTAMP '2017-04-19 22:00:00'
      GROUP BY TIME (HOURS(1) AND DISTRICTID,HOUSEID) USING TIMECODE(TD_TIMECODE) ) AS E_TSQ

WHERE W_TSQ.W_DID = E_TSQ.E_DID AND W_TSQ.W_GBT = E_TSQ.E_GBT
ORDER BY W_TSQ.W_DID, E_TSQ.E_HID, W_TSQ.W_GBT;
```

Weather Table
- Datapoint Approx every ½ hour
Electric Usage Table
- Datapoint Approx every 15 min

```
                  W_TCR                  W_DID   E_HID   AVG_WEATHER   AVG_ELEC_USAGE   MAX_ELEC_USAGE
’04-19-2017 08:00’, ’04-19-2017 09:00’     9      22        62.0           1000.0           1090.0
’04-19-2017 09:00’, ’04-19-2017 10:00’     9      22        65.0            999.0           1030.0
’04-19-2017 10:00’, ’04-19-2017 11:00’     9      22        68.0           1010.0           1100.0
’04-19-2017 12:00’, ’04-19-2017 13:00’     9      22        72.0           1030.0           1150.0
’04-19-2017 13:00’, ’04-19-2017 14:00’     9      22        73.0           1200.0           1330.0

’04-19-2017 21:00’, ’04-19-2017 22:00’     9      22        60.0           1000.0           1090.0

’04-19-2017 08:00’, ’04-19-2017 09:00’     9      44        62.0            750.0            800.0
’04-19-2017 09:00’, ’04-19-2017 10:00’     9      44        65.0            730.0            790.0
```

Source: Teradata

Figure 5

Temporal Data Types and Functions with Respect to Sensors

Temporal data types and SQL functions can be used with time series tables for in-database multi-variate time series analysis

In addition to time series tables, Teradata also supports temporal data types and temporal SQL functions, which can be very effective when combined with time series tables and IoT data. As an example, Teradata supports the PERIOD temporal data type enabling it to natively handle time periods. It also supports temporal SQL functions, such as OVERLAP, to understand temporal intersection.

An example of how this can be used with IoT data is Boeing Corporation who collect data from thousands of sensors in flight data recorders to diagnose aircraft malfunction. This includes mechanical indicators (landing gear, wing and tail flaps, cabin and baggage doors), cabin indicators (air pressure, temperature), and engine controls (fuel, speed, vibration, and pressure). Key challenges with this kind of data are that different sensors sample data at different rates, and there can be gaps when sensors do not emit data at all. Also, sensor readings in many cases are identical to the previous reading.

Boeing are using temporal data types and SQL functions to analyse flight sensor data to diagnose and predict aircraft malfunctions

These problems were overcome by a process called *temporal normalisation* where individually timestamped sensor readings are transformed into sensor readings that occur within a time period. In this context, a time period is defined as a period between a start time and end time during which a sensor reading remains constant, and there are no gaps in the data. Therefore, only one row is needed to represent all readings of the same value for a specific sensor for that time period. Boeing got several advantages from using this approach including:

- A 300-to-1 compression rate[4] in data to boost query performance as the same value for a sensor for a specific period was stored only once.
- Readings from multiple different sensors could be held in the same table.
- Gaps in the data were easily identifiable because a single row was created with a null value for a specific sensor for that period when no readings occurred.
- They could use temporal joins using the SQL OVERLAP function to look across multiple sensors for correlations and specific incidents in the data during the exact periods where the readings from those sensors intersected.
- Data could be loaded into temporal fact and dimension tables in their data warehouse and combined with other product and support data to analyse and visualise maintenance, scheduling, and more.

Scalable Dimensionality Reduction

Teradata also supports scalable in-database dimensionality reduction on time series data to eliminate noise prior to analysis

- With IoT data in Teradata time series tables, it is also possible to make use of the EXECR Operator for in-database execution of R code. This can be used for scalable dimensionality reduction to eliminate data not relevant to the types of analyses being conducted

[4] Made possible by the Teradata Database's NORMALIZE functionality

Script Operator

Teradata can also invoke Python scripts within the database to rapidly prepare IoT JSON data for analysis

- In addition to dimensionality reduction, Teradata also supports custom preparation of IoT time series data through its script operator. This enables in-database execution of custom Python code. With Python a programmer could unpivot IoT JSON data, converting it from columns to rows in the Teradata Database enabling analytical SQL functions to analyse it at scale

Geospatial 2.5D SQL

Support for geospatial indexing together with time series tables enables the analysis of data from mobile things

- Teradata also supports geospatial data types and geospatial SQL for analysis of mobile IoT data. In this way, time series tables can contain geometry-type columns representing the location of mobile devices at a specific timestamp. GeoSpatial indexes can then be placed on these mobile sensor co-ordinates and used in time series queries.

IoT and Teradata Unified Data Architecture™:

Besides the Teradata capabilities mentioned, IoT data processing and analysis can also be integrated into the wider Teradata Unified Data Architecture (UDA™) as shown in Figure 4. This shows a Teradata end-to-end capability that includes:

IoT data can be analysed in the Teradata database either on-premises or in the cloud

- Teradata Database support for the analysis of things both on-premises (using Teradata IntelliFlex®) and the cloud (using Teradata IntelliCloud™) or in a hybrid computing environment that spans both and that supports cloud bursting.

IoT data can be ingested into Hadoop and prepared before loading it into the Teradata Database for analysis

- The Teradata Hadoop Data Lake Appliance where time-series data can be ingested from the edge and loaded into Hadoop. Once in Hadoop, it can undergo data preparation before loading into the Teradata Database or the Teradata Analytics Platform for analysis. Also, cold time series data not often used could also be stored/archived in Hadoop. This would keep it on-line should a replay of a time series be required at some point in the future.

Sensor data in Hadoop can be accessed from and joined to data in a Teradata data warehouse via QueryGrid

- Teradata QueryGrid™ provides users with access to Hadoop time series IoT sensor data from a Teradata data warehouse as well as the opportunity to join IoT sensor data to data stored in a Teradata data warehouse.

IoT sensor data captured in NoSQL data stores or in cloud storage can be accessed and analysed from a Teradata data warehouse using QueryGrid and Presto

- Access to and analysis of sensor data captured in other data stores, such as Cassandra NoSQL database, Amazon S3 cloud storage or another database. This can be done from the Teradata Database using Presto federated query processing via the QueryGrid infrastructure. In this way, sensor data ingested into high-speed write processing data stores can be brought into Teradata Database and analysed.

Multi-variate time series analytic functions can run at scale on sensor data in Hadoop or in the Teradata Database

- Invocation of Teradata Analytic Platform functions to do multi-variate time series analysis on sensor data in Hadoop. These functions can run at scale in the Teradata Database or in-Hadoop and be invoked from Teradata via QueryGrid.

TERADATA IOT ANALYTICAL PROCESSING AT THE EDGE

Analytical models can be developed on the Teradata Database and deployed to run at the edge

Teradata has partnerships with several vendors to run analytics on their edge gateways

We have already discussed in Figure 1 that processing and analysing IoT data is needed at the edge as well as at the enterprise level on premises or in the cloud. In terms of processing and analysing IoT data at the edge, it is possible to develop analytical models on Teradata Database for deployment at the edge. This means that training data sets with edge-based IoT data are needed for development of supervised machine learning models on the Teradata Database and/or Teradata Hadoop Appliance so that models can be trained, tested and evaluated until ready for deployment. Typically, this is done using the Teradata Analytic Platform SDK analytic functions. Once these models are ready for deployment, Teradata has established a partnership with Dell to install them on their respective edge gateways products (for example, on the Dell Gateway 5000). This can be done using PMML. In addition, Teradata is supporting EdgeX Foundry. This is a vendor-neutral open source initiative hosted by The Linux Foundation that is building a common open framework for Industrial IoT edge computing. Teradata intends both Teradata Analytics Platform and Teradata Listener to work with EdgeX. In this way, centrally developed machine learning models can be deployed in a factory, in smart buildings, in equipment or other locations as part of the job of optimising business operations.

PROFESSIONAL SERVICES

In addition to technology, Teradata also offers professional services in the analysis of things. This includes capturing and preparing data, developing analytics to run at the edge and in the enterprise, as well as testing, deployment and monitoring. Teradata is offering professional services in several industry verticals including manufacturing, oil and gas, utilities, transportation, smart cities and automotive OEMs. Indeed, the Teradata Think Big Analytics team built a solution for a manufacturer to collect 30,000 sensor data files daily and ship them reliably to a data lake. This evolved into an open source self-service ingest, preparation and metadata service called Kylo. See http://kylo.io.

OVERALL BUSINESS BENEFITS OF TERADATA SOLUTION

Teradata support for time series offers significant productivity gains when preparing and analysing IoT data

Teradata has and continues to extend its products to support processing and analysing thousands of endpoints in the Internet of Things. These new capabilities provide significant productivity gains when dealing with IoT data capture and data preparation. The new time series tables with time buckets and time ordered storage significantly reduces time to value. Also, Teradata are doing all the heavy lifting in the database to easily ingest, store, organise and prepare data at scale to quickly make it available for analysis. In addition, they can bring the full power of the SQL language to bear on time series IoT data, offering semi-structured data types, temporal data types, temporal SQL functions, geospatial SQL and a large number of in-database and in-Hadoop advanced analytics. The value generated by combining the temporal capabilities and time series data is significant in that it allows data scientists to normalise data from multiple sensors into time periods, identify and remove gaps and perform multi-variate time series analysis at scale via SQL.

CONCLUSIONS

Time series data capture, preparation and analysis is now available at scale in the RDBMS

It is clear that Teradata is rolling out comprehensive support for analysis of the Internet of Things. The ability to support schema variant semi-structured data from edge gateways, three different classes of time series data, data partitioning and storage in timestamp order, scalable in-database data preparation, scalable analytical query processing, geospatial support for mobile things, multi-variate time-series analysis using temporal data types and functions and the ability to access time series data in NoSQL, Hadoop and Teradata Database is very comprehensive. In addition, they have accelerators to minimise effort on cleansing and preparing sensor data, to easily discover patterns of anomalies that frequently precede a failure, to optimise manufacturing performance and to detect anomalies in behaviour to help increase equipment availability/utilisation, improve safety, and reduce costs.   It is also possible to develop analytical models centrally, either on-premises or in the cloud, and deploy them in gateways at the edge.

Teradata also provide accelerators to expedite data preparation, discover anomalies and optimise manufacturing business operations

Any company looking to improve agility and move faster in the digital world by leveraging IoT data and analytics should consider Teradata if they are looking to significantly accelerate time series analysis.

About Intelligent Business Strategies

Intelligent Business Strategies is a research and consulting company whose goal is to help companies understand and exploit new developments in business intelligence, analytical processing, data management and enterprise business integration. Together, these technologies help an organisation become an *intelligent business*.

Author

Mike Ferguson is Managing Director of Intelligent Business Strategies Limited. As an independent IT industry analyst and consultant, he specialises in Big Data, BI/Analytics and Data Management. With over 35 years of IT experience, Mike has consulted for dozens of companies on BI/Analytics, big data, data governance, master data management and enterprise architecture. He has spoken at events all over the world and written numerous articles and blogs providing insights on the industry. Formerly he was a principal and co-founder of Codd and Date Europe Limited – the inventors of the Relational Model, a Chief Architect at Teradata on the Teradata Database and European Managing Director of Database Associates, an independent IT industry analyst organisation. He teaches popular master classes in Big Data Analytics, New Technologies for Business Intelligence and Data Warehousing, Data Virtualisation Enterprise Data Governance, Master Data Management, and Enterprise Business Integration.

Water Lane, Wilmslow
Cheshire, SK9 5BG
England
Telephone: (+44) 1625 520700
Internet URL: www.intelligentbusiness.biz
E-mail: info@intelligentbusiness.biz

EB-7306