Azure Data Explorer

Organisations these days are producing more data than ever before and are needing it to be analysed faster than ever before. This growth has been particularly pronounced for timeseries and telemetry data primarily generated by SCADA and IoT devices. These types of data have some unique characteristics – such as high-velocity, high-volume and the need for analysis and reporting in near real time – that can be challenging for many tools. The Azure Data Explorer (ADX) service is Microsoft's PaaS solution for ingesting, storing and analysing this type data. This article provides an overview of the capabilities of ADX for those that are new to the service.

The ADX service is based on a distributed database model than enables data ingestion and queries to be run in parallel across a number of clustered compute nodes. Data is stored in an optimised compressed column store database. The database is partitioned into a number of shards which are distributed evenly across the available cluster nodes. Each shard can contain several million rows which are encoded and indexed independently of other shards. Each cluster node caches data both in memory and on local SSD to further enhance query performance.

Users of the database are able to access data through tables consisting of rows and columns in a similar manner to relational database. However, it is not a fully relational database – as it does not define primary keys, foreign keys etc. Indexing of data is all handled automatically by the database. Materialised views can be created on the data, but standard views are not available. Functions can also be created that return either scalar (single value) or tabular results which can be used to simplify queries.

When designing a modern data platform or data architecture, ADX should be considered as an addition to the architecture to meet specific challenges required for near-real time reporting on timeseries and telemetry data, it is not intended to replace a data lake or a data warehouse. ADX is primarily designed for "hot-path" immediate reporting rather than long-term analysis. All tables in ADX have a retention period which defines how long data is to be kept in the database prior to being purged. Data in ADX should therefore also be ingested into a data lake where it can be accessed for long-term reporting and joined with other enterprise datasets.

Several methods are available for ingestion of data into ADX. The service provides data connectors for Azure Event Hubs, IoT Hubs and Event Grid which enables streaming data to be easily and rapidly loaded into the platform without the need for other tools. Event Grid connectors can be linked to data lake folders enabling automatic ingestion of data into ADX as it lands in the lake. Azure Data Factory can be used to bring in data from other data sources. Data can also be ingested programmatically into the platform using a variety of languages – e.g. .Net, Python, R, Java.

ADX has its own built-in dashboarding capability enabling reports, dashboards and charts to be quickly generated from data in the ADX database. Data in ADX can also be reported on using most common reporting tools. A native connector is available for Power BI. Tableau and most other reporting tools can connect using SQL Server ODBC drivers. Connectors are also available for common timeseries reporting tools such as Grafana.

Microsoft provide a number of client libraries that enable easy programmatic access to data in the ADX database – for example: .Net, Python, R, Java, Node and Go. A REST API is also available that enables access for other languages. A connector is also available for Apache Spark enabling Azure Databricks to process data in ADX.

Data in the ADX database can be queried using standard SQL SELECT commands. However, it is really designed to be accessed using a new language called Kusto. Kusto is based on the SQL language – so those of us who have spent their career mastering SQL have not wasted our time – however is more optimised, more flexible and contains a large number of enhancements specifically designed analysis of timeseries data. To help us SQL nerds learn the new language, Microsoft provide a Cheat sheet for SQL to Kusto translation. For example, a query using the ADX sample dataset to report on storm events in California could be written in each language as follows:

SQL Version

```
SELECT EventType, count(*) 'TotalEvents'
FROM StormEvents
WHERE State = 'CALIFORNIA'
GROUP BY EventType
ORDER BY EventType
```

Kusto Version

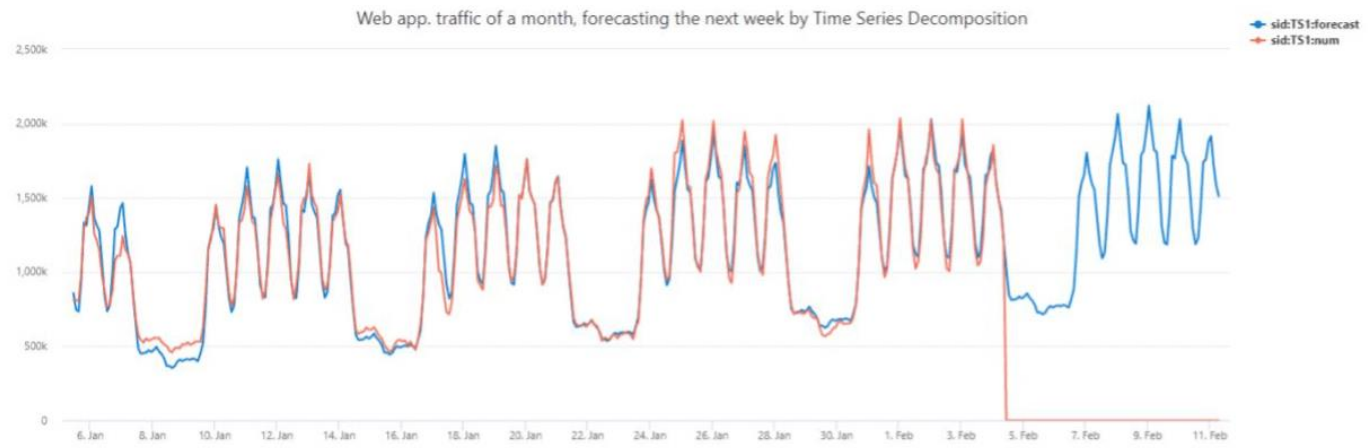
```
StormEvents
| where State == "CALIFORNIA"
| summarize TotalEvents = count() by EventType
| order by EventType asc
```

As a new language, Kusto also implements a large number of new functions that are not available in SQL. For example, with the addition of one line, the previous query can be presented as a barchart rather than a table:

```
StormEvents
| where State == "CALIFORNIA"
| summarize TotalEvents = count() by EventType
| order by EventType asc
| render barchart
```


Kusto also includes large number of language enhancements provide more advanced handling of time-series data. These include functions for managing data quality – e.g. by filling or interpolating values -, filtering, smoothing and analysing data. Statistical and machine learning capabilities are also built in to Kusto providing capabilities for anomaly detection, forecasting, linear regression, seasonality analysis amongst others. As an example (from Microsoft). the following creates a forecast of the next week of web traffic based on the previous month of data.

```
let min_t = datetime(2017-01-05);
let max_t = datetime(2017-02-03 22:00);
let dt = 2h;
let horizon=7d;
demo_make_series2
| make-
series num=avg(num) on TimeStamp from min_t to max_t+horizon step dt by sid
| where sid == 'TS1'   // select a single time series for a cleaner visualization
| extend forecast = series_decompose_forecast(num, toint(horizon/dt))
| render timechart with(title='Web app. traffic of a month, forecasting the next w
eek by Time Series Decomposition')
```



For organisations that have large amounts of timeseries data that they want to analyse rapidly and effectively, ADX can form a key part of the data architecture. If you would like further advice on how to implement ADX and manage timeseries data in your organisation, please contact a Fujitsu Data & AI specialist now.

Contact

Fujitsu Data & AI

+61 3 9924 3000