QAD Adaptive Applications
QAD Adaptive UX 2020.1

Implementation Guide

QAD Adaptive UX

70-3449-2020.1
QAD Adaptive Applications
QAD Adaptive UX
September 2020

QAD Inc.
100 Innovation Place
Santa Barbara, California 93108
Phone (805) 566-6000
http://www.qad.com

Table of Contents

QAD Adaptive UX Implementation Guide

This guide provides technical information for implementing various features of QAD Adaptive UX into an existing QAD Enterprise Edition environment.

Before proceeding with any installation or implementation tasks, be sure to read the release notes.

Product Name Changes

Starting in September 2019, the name for QAD's complete portfolio of products is QAD Adaptive Applications. Additionally, QAD Adaptive ERP is the new name for QAD's flagship ERP solution. QAD Adaptive ERP includes the functionality previously associated with QAD Cloud ERP and QAD Enterprise Applications - Enterprise Edition, plus the QAD Enterprise Platform and Adaptive UX, which resulted from the Channel Islands program. Going forward, the terms QAD Enterprise Applications, QAD Cloud ERP, and Channel Islands will be deprecated but will remain in previous documentation and training materials. QAD's intention is to—as soon as possible—eliminate the use of the deprecated terms going forward.

QAD Adaptive UX Installation

For QAD Adaptive ERP with QAD Adaptive UX, QAD performs and manages the installation. For on-premise installations of QAD Adaptive UX 2020, see the *QAD Adaptive ERP On-Premise Installation Guide*, available for early adopters in the QAD Document Library.

Recommended Web Browsers

The QAD Web UI is only supported on current versions of Chrome and Safari web browsers. Although other web browsers can be used, you may experience differing levels of performance and user experience.

To make sure you have the latest security updates, set your Chrome or Safari browser to receive automatic updates from Google or Apple.

For tablet use, the user interface is only supported on iPad Pro (or newer equivalent) with the Safari web browser. Although other tablets can be used, you may experience differing levels of performance and user experience.

Security Configuration

QAD Enterprise Platform includes comprehensive security features. For more information about security, please see the *QAD Security Administration Guide*, located in the QAD Document Library.

Role Permissions Configuration

Roles are used to model the business processes that exist within a business enterprise. Roles determine the set of application resources that display for users when they access their permitted workspaces. In order to model your organization's business processes effectively, users need access to all the appropriate application resources required for them to perform their everyday business tasks.

For more information about user and role configuration, read the *QAD Security Administration Guide*, *Users and Roles* (chapter 6), located in the QAD Document Library.

Do Not Change Permissions of QAD Default Roles

QAD Adaptive UX arrives with a variety of predefined, pre-configured roles. These roles are provided as starting points for the roles you will create for your system. You should not use the QAD Adaptive UX default roles, except for webui_user, for your users because any permission changes you make to the default roles will be overwritten with each software update. You can review the default roles' associated permissions on the Role Menus and Role Permission screens and use these default settings as a guide when assigning permissions to new roles.

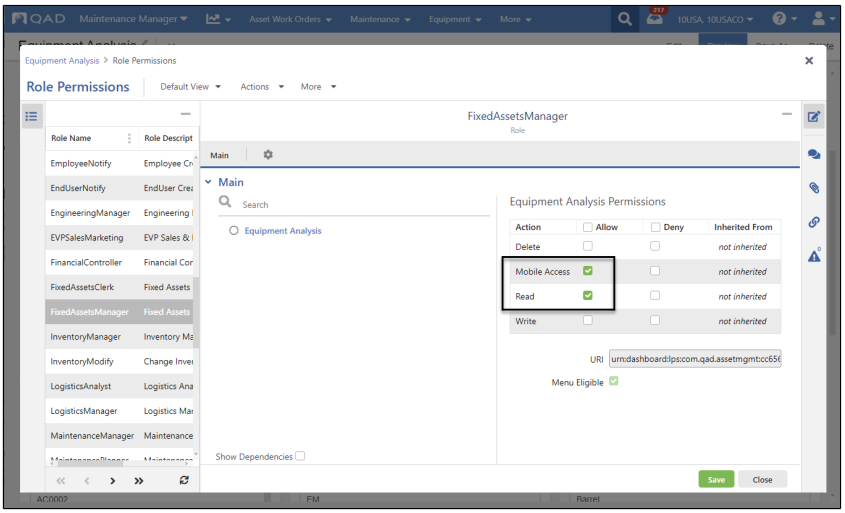
Action Center Mobile App Permissions

System administrators, and other users whose roles have adequate authority, can provide users access to the Action Center mobile app from an action center itself or from **Role Permissions**. QAD recommends granting access directly from the action center for ease of use.

Granting Access from an Action Center

1. Navigate to the action center.
2. Select Permissions from the More menu.

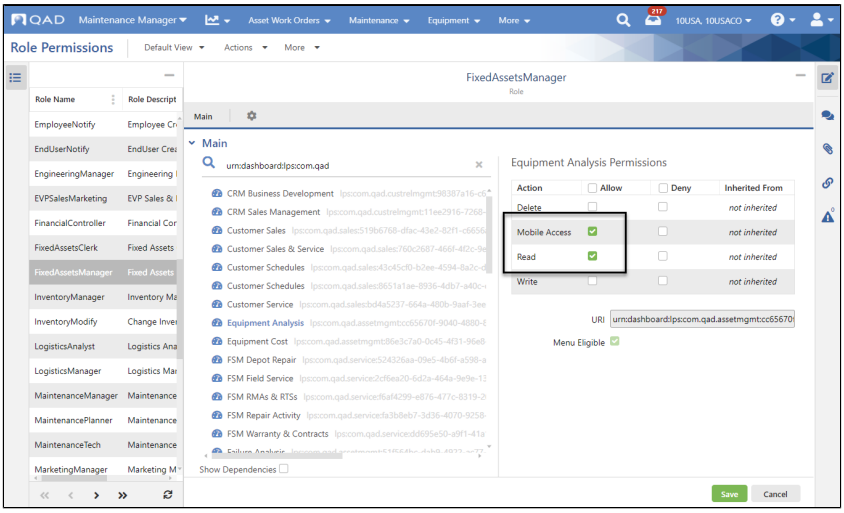
3. Open the user's role.
4. In the Permissions grid, select Allow for the Mobile Access and Read options.



5. Select Save.

Granting Access from the Role Permissions Screen

1. Navigate to **Role Permissions**.
2. Open the user's role.
3. Find the action center in the resource tree. Each action center is a separate resource stored in an app in the Apps list in the Role Permissions resource tree. The URI of each action center starts with “urn:dashboard:lps:com.qad”. To locate a particular predefined QAD action center, search for the URI stem (urn:dashboard:lps:com.qad) or search for the app that relates to the action center’s functional area; for example, the Financial Analysis action center is stored in the Financials app and its URI is urn:dashboard:lps:com.qad.financials:7c842c96-b67d-4094-aef0-8c3b71804fad.



4. In the Permissions grid, select Allow for the Mobile Access and Read options.
5. Select Save.

Quality Orders - Hiding the Specification and Specification Detail Fields

In Item Attributes and Quality Control (IAQ), the Specification and Specification Details fields can be hidden in grids on **Quality Orders** for specific roles. This is useful if you do not want a certain role to be influenced by the specification information when entering attribute values.

To hide the specification fields in every browse and grid in **Quality Orders**, you will need to use **Role Permissions** and the Display Specification for Results Entry options in **Inventory Control** and **Quality Control.** See the following table for more information:

Screen	What is Affected	Notes
Role Permissions	Quality Orders (Lot and Quality Type) • Quality Orders > Attributes panel (grid and details form) • Quality Orders > Test Records > Test Attributes panel (grid and details form)	The Specification and Specification Details fields are hidden for the specified roles that do not have read access.
Inventory Control > Display Specification for Results Entry field	Quality Orders (Lot Type) • Quality Orders > Attributes Details screen > Attributes browse	The values in the Specification and Specification Details fields are blank for all users, regardless of role.
Quality Control > Display Specification for Results Entry field	Quality Orders (Quality Type) • Quality Orders > Test Records Details screen > Test Attributes browse • Quality Orders > Attributes Details screen > Attributes browse	The values in the Specification and Specification Details fields are blank for all users, regardless of role.

Hiding the Specification Fields

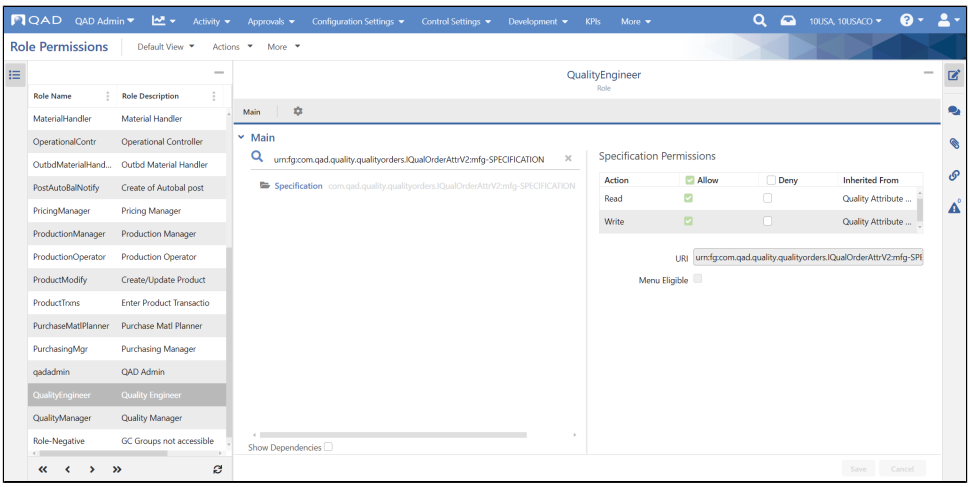
You can hide the Specification and Specification Details fields directly through **Role Permissions** or by using the role permissions functionality in the **Quality Orders** screen.

Hiding the Specification Fields From Role Permissions

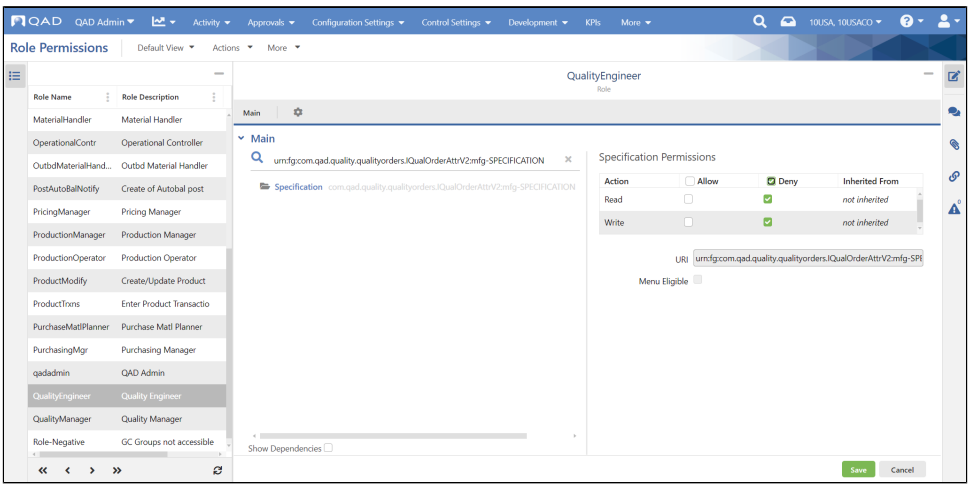
Follow these steps to use **Role Permissions** to hide the Specification and Specification Details fields:

1. In **Role Permissions**, select the role for which the specification fields will be hidden.
2. In the Search field, enter the URI for the components that will be affected:

Component /Screens	URI	Notes
Quality Orders > Attributes panel	urn:fg:com.qad.quality.qualityorders.IQualOrderAttrV2:mfg-SPECIFICATION	Hides the Specification and Specification Details fields in the Attributes panel (grid and details form) for quality orders (lot and quality type).
Quality Orders > Test Records > Test Attributes panel	urn:fg:com.qad.quality.qualityorders.IQualOrderTestAttr:mfg-SPECIFICATION	Hides the Specification and Specification Details fields in the Test Attributes panel (grid and details form) for quality orders (quality type).



3. To hide both fields, select Deny for Read and Write for the two URIs listed in the table.



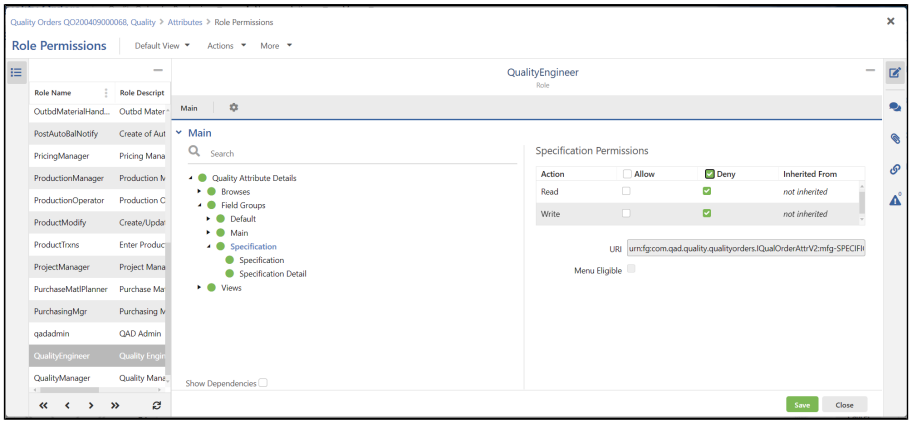
4. Select Save. The specification fields will now be hidden in the **Quality Orders** > Attributes panel and **Quality Orders** > Test Records > Test Attributes panel.

Hiding the Specification Fields From Quality Orders

Alternatively, you can hide the the Specification and Specification Details fields by navigating through the **Quality Orders** s creen.

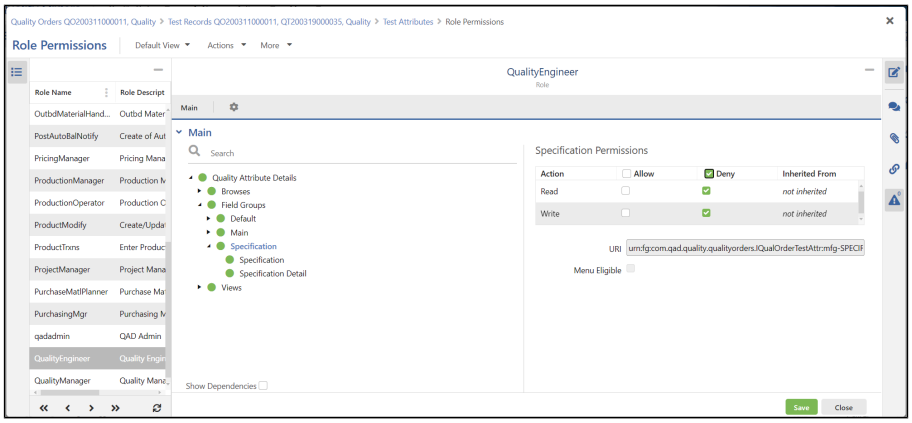
Follow these steps to hide the specification fields:

1. Hide the specification fields in the Attributes panel for quality orders (lot and quality type):
    a. In **Quality Orders**, select a quality order (lot or quality type) and open the Attribute details screen.
    b. From the More drop-down menu, select Permissions.
    c. In the Role Permissions screen, select the desired role.
    d. Then select Quality Attribute Details > Field Groups > Specification. The Specification and Specification Details fields are grouped together under the Specification field group. To hide both fields, select the Specification field group and then select Deny.



e. Select Save.
f. For the specified role, the Specification and Specification Details fields will now be hidden in the Quality Orders > Attributes panel.

2. Hide the specification fields in the Test Attributes panel for quality order (quality type):
   a. In **Quality Orders**, select a quality order (quality type) and open the Test Records > Test Attributes details screen.
   b. From the More drop-down menu, select Permissions.
   c. In the Role Permissions screen, select the desired role.
   d. Then select Quality Attribute Details > Field Groups > Specification. The Specification and Specification Details fields are grouped together under the Specification field group. To hide both fields, select the Specification field group and then select Deny.



e. Select Save.
f. For the specified role, the Specification and Specification Details fields will now be hidden in the Quality Orders > Test Records > Test Attributes panel.

Custom Browses and Drill Downs

Browse Naming

As an administrator, you can define custom browses using Browse Maintenance in the QAD .NET UI. These browses are available in the QAD Web UI when the qra-sync YAB command (yab qra-sync) is run.
When defining custom browses, the labels for menu items on the QAD .NET UI are defined in Menu System Maintenance (in the Label field), while the labels for menu items in the QAD Web UI are defined in Browse Maintenance (in the Description Term field).

When defining a custom browse, be sure that the name of the browse (the label) is unique to avoid user confusion. Note that the browse naming convention on the QAD .NET UI is to have the name of the browse end with "Browse," while on the QAD Web UI, the menu item type is indicated by the menu type icon. Do not include "Browse" at the end of the name.

To ensure consistency, the Menu System Maintenance Label field setting should match the Browse Maintenance Description Term field, but do not include "Browse" at the end of the QAD Web UI's string.

Drill-Down Link Definitions

As an administrator, you can define drill-down links on the QAD .NET UI using Drill-Down/Lookup Maintenance.

For the drill-down links to be included on a QAD Web UI hybrid view, you first need to identify the browse used by the hybrid view of interest. To identify the browse used by a hybrid view, locate the browse on Role Permission Maintain's Secured Resources tab and get the browse identifier. For example, the browse used with the Sales Orders hybrid view is identified as "so803.p".

With the browse identifier, you can then define the drill-down links for it using Drill-Down/Lookup Maintenance. Note that in Drill-Down/Lookup Maintenance, in the Calling Procedure field, the browse identifier includes "br". For example, for "so803.p", in the Calling Procedure field, you enter "sobr803.p". You can then specify the Procedure to Execute.

Once the link has been defined in Drill-Down/Lookup Maintenance, the change is included on the QAD Web UI after an administrator runs the following YAB commands:

- Update resource dependencies: `yab qra-resource-dependency-update`
- Restart Tomcat: yab tomcat-webui-stop tomcat-webui-start

Action Centers

This section describes the architecture and components supporting the Action Centers, along with instructions for installing, configuring, and troubleshooting them.

It does not comprehensively cover installation or administration, but highlights points specific to the Action Centers and refers to other guides as needed.

Logi Analytics Technical Overview

The Action Centers are implemented using analytics tools and frameworks provided by **Logi Analytics**, a QAD partner. Logi software supports the creation, maintenance, and display of the Action Centers and their contained visuals, which can be completed by trained users as a self-service activity not requiring additional software development. Visuals (charts, tables, gauges) can be created and modified in an iterative manner, shared with other users and QAD roles through a common gallery, and included in Action Centers by QAD users in roles with the correct permissions. Logi components have been embedded into the Web UI and integrated with the QAD data sources – mainly KPIs and their underlying browses.

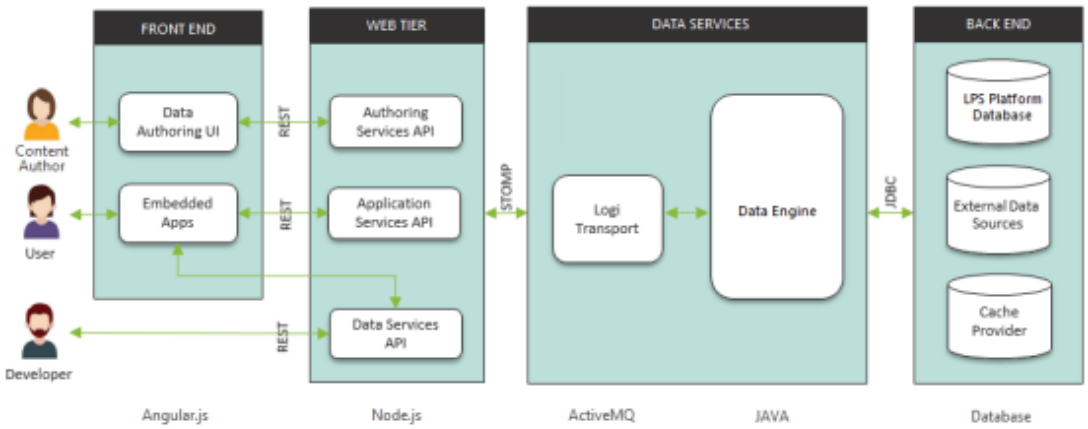
Two Logi Frameworks

Logi provides two frameworks for the development and display of graphical analytics: **Logi Info** and **Logi Platform Services** (LogiPS). **Logi Info** is the older framework, and was the only one used in Channel Islands before the September 2019 release. **Logi Platform Services** was introduced in the September 2019 release. *By default, customers installing or upgrading to the September 2019 or later releases can create Action Centers using Logi Platform Services only.* Logi Info is also provided, but is enabled for read-only use so that existing Action Centers provided by QAD or built by the users can still be displayed. In addition, as of the September 2019 release, all KPIs created in the KPI screen of the Web UI automatically use Logi Platform Services for creating and displaying visuals. The visuals for existing KPIs that were created using Logi Info can be displayed but not changed. Instead, the KPI must be copied to a new KPI that uses Logi Platform Services and the visuals re-created using the Logi Platform Services functionality.

The technical overview describes the high-level architecture of both Logi Platform Services and Logi Info.

Logi Platform Services

Logi Platform Services (LogiPS) uses a multi-tiered, service-oriented architecture to deliver rich visualizations and dashboards (Action Centers) in a more compact and modern form than other products.



The four components of the LogiPS architecture include the Front End, Web Tier, Data Services, and Back End.

LogiPS provides system administrators and content creators with an easily secured and configured system that can be embedded in web sites and applications without the use of iFrames. Visualizations, dashboards, and reports are created and stored in libraries, and the required code for embedding them is generated on request.

LogiPS uses an advanced data retrieval technology based on the *Dataview*. A Dataview is defined by a JSON document that specifies connection information, query details, and data enrichment options. This Dataview Definition (DVD) is stored in a system database and executed at runtime.

When executed at the server, a Dataview retrieves data, caches it in a self-tuning columnar data store, and makes it available for use with visualizations. The benefits of this include:

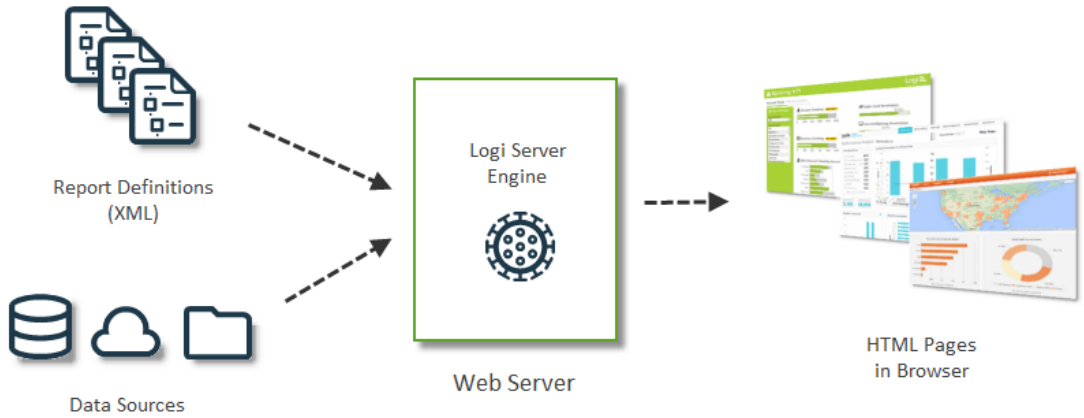
- Decoupling of the data and the visualizations
- Dataviews can be shared among multiple users
- Dataview changes manifest themselves immediately everywhere the Dataview is used

- Dataviews provide excellent performance for very large datasets (250M+ rows)

For Action Centers, the data is retrieved from the Query Service with SQL queries, as referenced in later sections of this document.

Logi Info

Logi Info is a framework for developing and displaying analytical data embedded inside a host application, in this case QAD Adaptive Applications. The resulting functionality is packaged and deployed as a web application, rendering visualizations and serving them to the browser as HTML pages.



A Logi application consists of web page sources known as 'report definitions.' Some of these pages provide robust self-service features that allow end users to define their own charts and cross-tabulation grids interactively, given the source data and metadata. These visuals can then be published and added to user-defined dashboards (Action Centers).

Logi Info applications separate the development, data access, and presentation processes, as shown in the graphic.

1. *Report Definitions* are text files that contain the information describing report layout and contents, stored as XML documents. While it is possible to create and edit definitions with any text editor, Logi Studio provides an integrated development environment with tools and helpful wizards that do much of the coding for you, reducing development time and effort.
2. When a report page is requested by a user, the Logi Server Engine**,** on the web server, processes the report definition and accesses whatever *data sources* are required. A wide variety of data sources are supported and data caching is used to speed up performance.
3. The Logi Server Engine formats the retrieved data and presentation details based on the definition and accompanying style sheets, generates *HTML* and *JavaScript*, and returns the report page to the user's browser for viewing.

There are two main data sources accessed by Logi Info to populate the Action Centers.

1. *Browses*: The same kinds of browses that can be defined by end users and displayed from the standard menu can also be used to retrieve data for the Action Centers through APIs called by Logi Info. The data sets consumed by Logi Info consist of the records returned by the browses.
2. *Financials Report Writer (FRW)* KPIs: The Financials Report Writer allows financial users to define key performance indicators using the enterprise's financial data, chart of accounts, and reporting structure. This information is also provided to Logi Info through APIs.

Query Service Technical Overview

This section describes the high-level technical architecture of the Query Service, a component that returns the business data required to display the Action Centers. The Query Service was introduced in the September 2018 release of Channel Islands.

Key Concepts

Reasons for Change

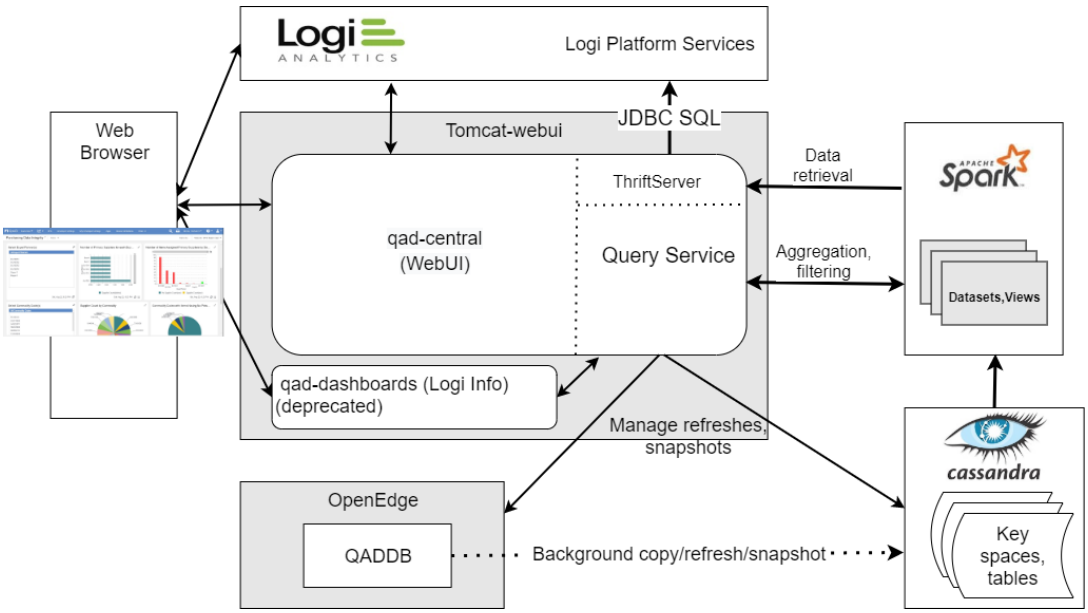
Previous Action Center releases had performance limitations when summarizing and displaying large amounts of source data, principally the very large result sets returned by some browses. In particular, performance degraded rapidly as the number of records returned by a browse increased. To mitigate this problem, the source data that could be included in Action Centers was limited to 5,000 records or less. The performance limitations have several causes.

- The overhead of processing browse requests in Progress AppServer agents, which often take a long time to complete and have high CPU usage. AppServer-based solutions are therefore difficult to scale up for high-volume environments.
- The high memory usage and slow processing time required in the Action Center web application to load and display very large data sets. The Logi Info software powering the Action Centers is optimized for self-service visualization and rendering, not high-volume data grouping and aggregation.

To address these issues, the architecture supporting data queries, post-processing, and retrieval was implemented using a Query Service.

Query Service

The original browse data retrieval engine was supplemented and partially replaced by a new infrastructure known as the Query Service. The Query Service incorporates several concepts and third-party components to bring required source data into the Action Centers more quickly and in a more compact, usable form.



SQL-Based Retrieval of Business Component Browses

The Query Service supports the use of virtually all kinds of QAD browses as data sources for the Action Centers.

- MFG (operational) browses
- FIN (financial) browses
- BC (business component) browses

In the case of BC browses, infrastructure is provided to retrieve the OpenEdge source data through JDBC connections, rather than ABL code running in a Progress AppServer. This approach significantly reduces the CPU overhead and processing time to retrieve the browse data. It also leverages open SQL and JDBC standards, as opposed to the more closed and proprietary Progress AppServer architecture.

While current QAD-defined KPIs are based on MFG and FIN browses that rely on the older AppServer-based browse engine, in future releases this is expected to change as user-defined business components are more widely used and legacy browses are migrated to the newer infrastructure with SQL-based retrieval.

Data Lake

Data feeding the Action Centers is copied from its OpenEdge sources into a separate data lake repository built on Apache Cassandra. Cassandra stores the information in de-normalized form, where the relational data is joined and flattened before being written. Cassandra's columnar data structure is optimized for immutable (read-only) storage and fast retrieval, unlike relational databases such as OpenEdge, which are designed to support general-purpose CRUD operations.

For Action Centers, the data stored in the data lake is refreshed from its OpenEdge source overnight by default, when scheduled refresh is enabled both at the system level and for individual KPIs. As previously mentioned, the retrievals are processed using a combination of JDBC-based queries and Progress AppServer agents, depending on the browses needed. Refreshes may also be requested manually by end users for individual KPIs in Action Center panels.

Beginning with the September 2020 release, the data lake is also used to store historical snapshots of KPIs, for those KPIs that are defined as 'historical' instead of 'current.' This functionality allows KPI results from different points in time to be presented together for comparison and trend analysis purposes. The frequency of the snapshots and the total number allowed in the data lake are limited in order to manage disk space effectively, but are configurable by KPI. Unlike the scheduled refreshes of current KPI data mentioned previously, the historical snapshots are not re-createable from the OpenEdge sources. For historical KPI snapshots, the data lake is therefore the system of record.

In future releases, the data lake is planned to support other functions outside of Action Centers including historical archiving, reporting, and business intelligence. Its use will not be confined to Action Centers. However, supporting the Action Centers is its role in the current release.

In-Memory Post-Processing

Browse data stored in Cassandra is cached in memory using Apache Spark. Spark is a fast, scalable, in-memory data processing layer that can respond to queries in a flexible way, using any database fields as indexes. It also applies filtering, grouping, aggregation, cross-tabulation, and deduplication on demand much faster and more efficiently than could be done by OpenEdge ABL code running in a Progress AppServer.

At runtime when an Action Center is displayed, the data can be pre-grouped and pre-aggregated by Spark based on the definition of the KPIs associated with that Action Center. The data returned to the Action Center is much more compact as a result with fewer records, allowing the Action Centers to overcome the 5,000-record limit for source data prescribed in previous releases.

Optimized Logi Integration

Older Action Center releases leveraged the Logi Info framework from Logi Analytics as the basis for Action Center definition and display. The Logi-based Action Center artifacts are deployed as a separate web application, named qad-dashboards by default, inside the tomcat-webui server. This web app is tightly integrated with the primary QAD Web UI web application, with Logi screens embedded inside the Web UI window and menu. REST API calls passing between the two web apps are used to communicate user directives, metadata, and business data into the Logi environment.

Beginning with the September 2018 release, the Logi integration was enhanced to take advantage of the superior performance of the new Query Service. As end users design visuals and their data contents using the Action Center's self-service capabilities, metadata defining the group-by categories and numeric aggregations needed to support those visuals is automatically extracted from Logi and sent to the Query Service. The Query Service is then able to apply the maximal a mount of grouping and pre-aggregation that Logi can consume in order to render the requested charts. In summary, the Logi integration optimizes its request to obtain exactly the required data in pre-processed form, with the result set made much smaller through Query Service grouping.

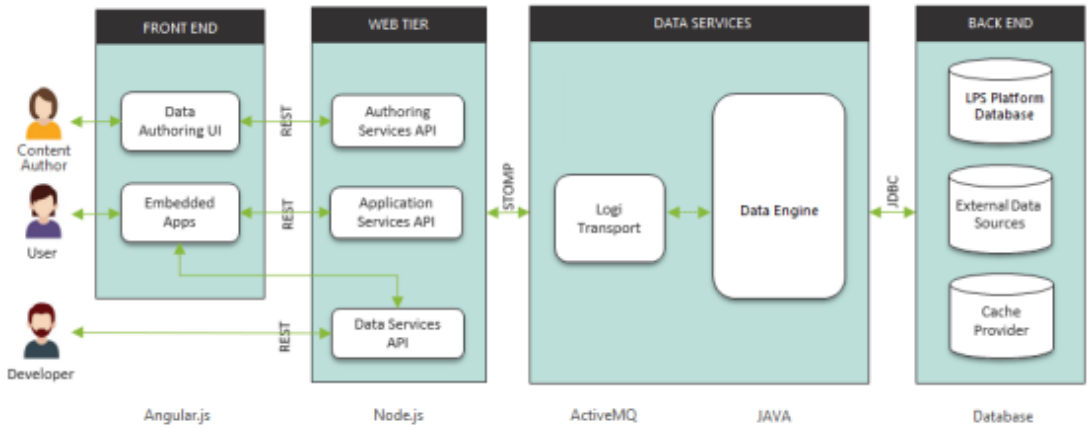
Beginning with the September 2019 release and the introduction of Logi Platform Services (LogiPS), the Logi integration was re-engineered again to allow Logi to retrieve the Query Service data using SQL through a JDBC connection, which is supported inside the Query Service by the Spark ThriftServer. This approach allows Logi to retrieve the data as though from a relational database, which better leverages the built-in capabilities of LogiPS to group and aggregate the data automatically to suit the needs of each visual.

Action Center Key Components

The Key Components section describes the major third-party components that are critical to the Action Center architecture.

Logi Platform Services

Logi Platform Services (LogiPS), a.k.a. Logi Composer, uses an n-tier, service-oriented architecture that layers presentation services, application services, and data services into distinct sets that can be deployed logically and physically at different tiers in an environment.



The four components of the LogiPS architecture include the front end, web tier, data service, and back end.

In a front end client, a developer embeds the desired Logi widgets into an HTML web page or application. Widgets are configurable, client-side components that are used to create visualizations. Multiple widgets can be embedded in a web page and they can communicate with one another. Content authors and users will use the client to interact with LogiPS. The complex widget used to build visuals is called a **Thinkspace** view.

The front end runs on Angular.js, which supports the model-view pattern on the client. Angular uses Restangular, a service that handles REST API requests, to communicate with Logi application and data services. Responses are delivered with JSON data.

The web or **application service** tier passes the HTTP requests to the appropriate handler on the server. It uses Node.js, an embedded, run-time environment for server-side web applications that handles all server-side processing at the application level. Node.js uses the Express framework, which is a lightweight web server with full REST support.

The LogiPS **data service** processes and queues requests from the application service using the ActiveMQ message broker, communicating via the STOMP protocol. ActiveMQ passes the request to the Logi Transport service, which interprets it and makes a request to the Logi Data Engine for retrieval of the requested data.

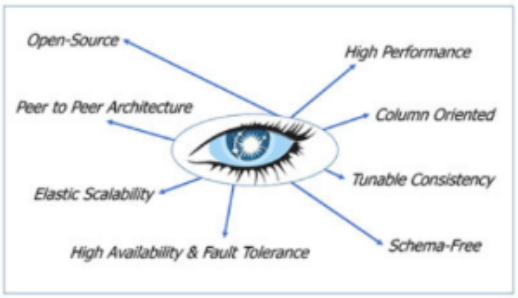
LogiPS uses the back end via JDBC to retrieve and process the appropriate Dataview (stored in the Platform Database), which in turn retrieves the appropriate data from an external data source or cache.

The server-side components are run within two separate processes, which are started and stopped separately.

- **Data Service**: Includes the data services tier, containing the data engine and controlling the LogiPS Platform Database (PDB).
- **Application Service**: Includes the web tier and the REST APIs called from clients to maintain the contents of the PDB.

The self-service capabilities of Logi Platform Services are enhanced to consume larger volumes of source data in a more performant way using the Query Service's other components, which are described in the following sections.

Apache Cassandra



Not a Relational Database

Cassandra is not a relational database intended for transaction processing. Rather, it is a multi-node, horizontally scalable, columnar database with an embedded SQL-like language. It is used as a standard data lake repository by many of the largest companies in the world due to its speed, reliability, distributed architecture, and flexible data model.

In small- to medium-sized environments, Cassandra will be deployed on the same node as the core Enterprise Application Infrastructure. In larger environments, it may reside on a dedicated server as part of a larger data lake infrastructure. In very large, multi-site enterprise environments, Cassandra could be deployed as a decentralized cluster with multiple nodes.

While Cassandra is a columnar database, many of the key concepts with which application developers and DBAs are familiar also apply to Cassandra. There are tables, records, and fields. However, data is internally organized into columns rather than rows, unlike relational databases such as OpenEdge and Oracle. This makes Cassandra well suited to high-volume queries and analytics, where users often drill down by columns (for example, "Show me all the data for this item or this customer") rather than transactions, as in traditional ERP (for example, "Create an invoice for this customer").

Columnar databases support a flexible schema model and much improved performance for analytics, but do not support join operations or secondary indexes very well. In implementations that require the flexibility of user-defined joins and filters, these actions should be done in complementary frameworks such as Apache Spark (see below).

Some advantages of Cassandra are:

- High availability
  - Distributed table storage
  - Tunable consistency
  - Fault tolerance
- High performance
  - Low latency
  - Linear scalability
  - Table-specific tuning
- Data model flexibility
  - Dynamic schema controlled by queries
  - Management and monitoring via JMX and SQL queries
- Open source licensing

Terminology

The following basic terms are necessary to understand how Cassandra works.

- Cassandra is a distributed database running nodes in a *cluster*. The nodes communicate in a peer-to-peer, master-less fashion.
- Cassandra rows are stored in *tables*, where each table has a mandatory primary key.
- A *keyspace* groups tables as a logical entity, similar to a schema in relational databases. In the Query Service, all browse result sets are stored in the "browses" keyspace.
- Data is accessed via *CQL*, an SQL-like query language.
- Cassandra writes to a data log first, similar to the OpenEdge before-image file. It then writes to an in-memory cache inside a JVM heap called *memtables* before flushing to disk (SSTables).
- Cassandra housekeeps the data on disk, compacting it and discarding *tombstones*, which are markers placed inside obsoleted data to mark it for later physical deletion.

The following table cross-references Cassandra terms to similar concepts from OpenEdge.

Cassandra	OpenEdge
Cluster	N/A
Column	Field

CQL	ABL
Data Center	Replication Group
Data Log	Before-Image File
Flat Data	Relational Data
Key Cache	Buffer Pool
Keyspace	Database
Memtable	Buffer Memory
Node	Server
Row (single record)	Row (unit of replication)
SSTable	Record block
Table	Table
View	add another index

Compaction and Deletion

Cassandra collects all the versions of a row, and from them assembles the most up-to-date versions of that row. It then writes the new row versions to a new SSTable, leaving the old versions along with other rows that are ready for deletion in the old SSTables. As soon as all pending reads are completed, Cassandra deletes the old versions using markers called tombstones, which indicate that the data has been obsoleted.

After many deletes, the resulting tombstones can grow to consume a significant amount of disk space and slow Cassandra processing. However, Cassandra deletes tombstones automatically in its compaction runs, which are triggered every few minutes. By default, tombstones older than 10 days are deleted during compaction. While this time period can be configured if needed, in the case of Action Centers this should not be necessary. The browse data stored in Cassandra is only deleted during a scheduled or manual refresh, and the refresh causes the affected Cassandra tables to be dropped and re-created. Hence, individual rows are not deleted and tombstones will not accumulate in the database.

Cassandra performs best with local low-latency SSD or SAS storage, which is also cheaper to provision than relatively high-latency network storage. It uses an efficient log–structured engine that converts updates into sequential I /O. Cassandra's storage engine does not read or rewrite existing data when processing updates, but only appends the updated data. This approach allows updates to be processed very fast. However, updates and deletes can be expensive and generate tombstones, which can affect query performance.

Core Tools

Several native Cassandra tools are useful for system administrators and DBAs. Basic database start, stop, remove, rebuild, and other functions are controlled through YAB and not listed here. To see a description of the YAB commands for managing Cassandra, use the command 'yab help cassandra-'.

- SSTable: Table utilities such as dump, print metadata, split table, list tables.
- nodetool: Comprehensive utility used to monitor and manage a cluster. This tool can also be started using the 'yab cassandra-default-nodetool' command.
- cqlsh: Command-line utility for connecting to a Cassandra database and executing CQL commands.
- cassandra-stress: Stress testing tool.

Some of these tools are implemented in Python, but Cassandra in general is entirely Java-based.

Memory Mapped Files

Cassandra uses memory-mapped files (mmap) internally. That is, the operating system's virtual memory is used to map a number of on-disk files into the Cassandra process address space. This uses virtual memory or address space, and is reported by O/S tools, such as top, accordingly. However, on 64-bit systems virtual address space is effectively unlimited, so it is seldom a concern.

A key point is that for a mmap'd file, there is never a need to retain the data in physical memory. Thus, whatever is in physical memory is there only as a temporary cache, in the same way that normal I/O will cause the kernel page cache to retain data that has been read or written.

The major difference between normal I/O and mmap is that in the mmap case, the memory is mapped to the process, thus affecting the virtual size as reported by top. The main argument for using mmap instead of standard I/O is that read actions only need to access memory; there is no page fault, therefore no kernel overhead to perform context switching.

CAP Theorem and Cassandra

The well-known CAP theorem states that it is impossible for a distributed computer system to simultaneously provide all three of the following guarantees.

- *Consistency*: All nodes see the same data at the same time.
- *Availability*: Every request receives a response about whether it succeeded or failed.
- *Partition* tolerance: The system continues to operate despite arbitrary message loss or component failure.

All databases can be categorized as CP, AP, or CA, based on which of the guarantees are supported. For example, OpenEdge and other relational databases are CA databases, while MongoDB and ElasticSearch are CP databases. Cassandra is an AP database. The advantages of an AP database are greatest in environments where short periods of data inconsistency are preferred over short periods of database unavailability.

Cassandra satisfies a weaker consistency requirement by adopting the BASE standard, which is a modified version of the ACID (Atomicity, Consistency, Isolation, Durability) properties satisfied by most relational databases.

- *Basically Available*: The system guarantees the availability of data in the sense that it will respond to any request. However, the response could be a failure to obtain the requested data, or a data set in an inconsistent or changing state.
- *Soft*: The state of the system is always “soft” in the sense that eventual consistency, described below, may cause changes in the system state at any given time.
- *Eventually Consistent*: The system will eventually become consistent once it stops receiving new data inputs. As long as the system is receiving inputs, it does not check the consistency of each transaction before it moves to the next transaction.

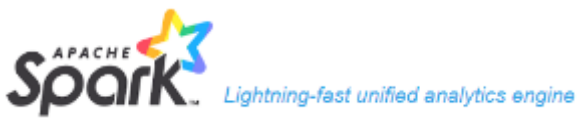
Full consistency has a negative effect on cost-effective horizontal scaling. If the database needs to check the consistency of every transaction continuously, a database with billions of transactions will incur a significant cost to perform all the checks. The idea of consistency is not practical in a large distributed database. It is the principle of eventual consistency that has allowed Google, Twitter, and Amazon, among others, to interact with millions of their global customers, keeping their systems available by supporting partition tolerance. Without the principle of eventual consistency, today's systems could not support the exponential rise of data volumes caused by cloud computing, social networking, and related trends.

Limitations

Cassandra is in some ways very restrictive compared to relational databases like OpenEdge, as summarized below.

- *Joins*: Cassandra does not allow joins, and is therefore not suitable for representing normalized data models. Join s must be implemented in a separate component such as Apache Spark, which is described in the following section. The data stored in Cassandra should be self-describing documents (for example, an invoice object in XML or JSON form) or de-normalized, flattened views designed for query purposes.
- *Transactions*: Cassandra supports only "lightweight" transactions, essentially only existence checks, without the ACID compliance common in relational databases.
- *Secondary Indexes*: Cassandra does not fully support secondary indexes, as it imposes heavy restrictions on which fields can be included in a secondary index.
- *Text Search*: Cassandra allows full text search with advanced features, but only on specific fields with a text search index.
- *Tombstones*: Cassandra does not delete and update data in real time like a relational database. In order to ensure good read performance, it simply marks the affected data with a tombstone. Future queries automatically skip over the tombstones. However, tombstones can build up quickly in tables with heavy delete/update activity, even to the point where there are more tombstones than actual data. In such cases serious performance problems can result, as reading tombstones can cause excessive latency, timeouts, and even exceptions. Tombst ones also consume disk storage unnecessarily.

Apache Spark

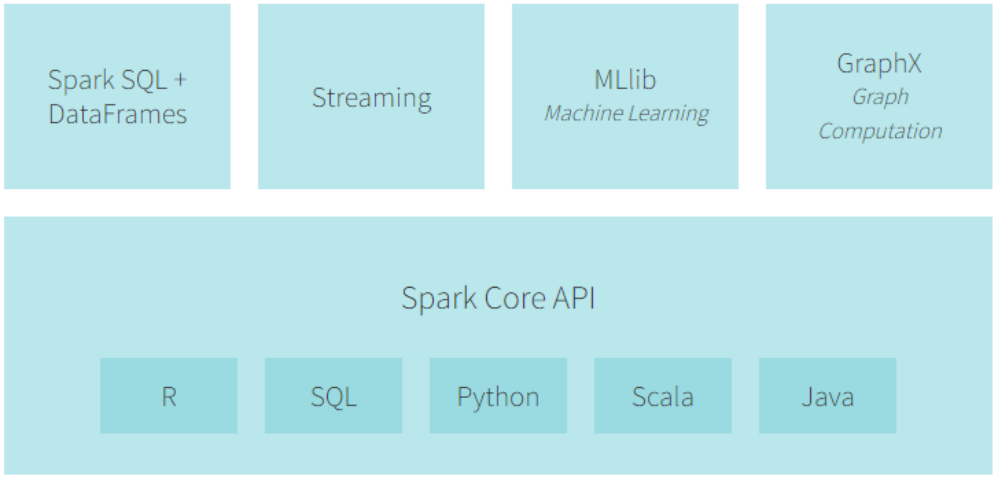


Apache Spark is a general purpose in-memory data processing engine and cluster computing framework that can perform Extract, Transform, and Load (ETL) operations, ad-hoc queries, machine learning, and graph processing on large volumes of data at rest (batch processing) or in motion (stream processing).

It supports native APIs for manipulating and querying data in the following programming languages: Scala, Java, Python, R. In addition, it provides libraries that allow the same data to be accessed through more specialized and advanced languages and protocols.

- *SQL*: A Spark module for structured data processing. It provides a programming abstraction called DataFrames to access data organized into named columns, like a relational table, and can act as a distributed SQL query engine.
- *Streaming*: Spark Streaming is a scalable fault-tolerant streaming system, receiving data streams and chopping them into batches. Spark then processes those batches and pushes out the result. Besides working directly with files and sockets, it integrates with a variety of popular data sources, including HDFS, Flume, Kafka, and Twitter.

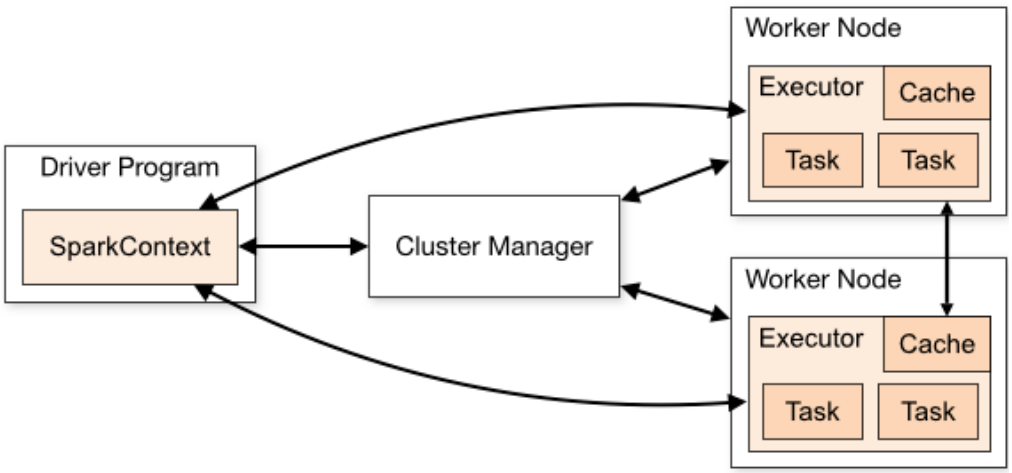
- *MLlib*: Built on top of Spark, MLlib is a scalable machine learning library that provides high-quality algorithms performing at high speeds. The library is usable in Java, Scala, and Python.
- *GraphX*: A graph computation engine built on top of Spark that enables users to interactively build, transform, and reason about graph structured data at scale. It comes with a library of common algorithms.



Architecture

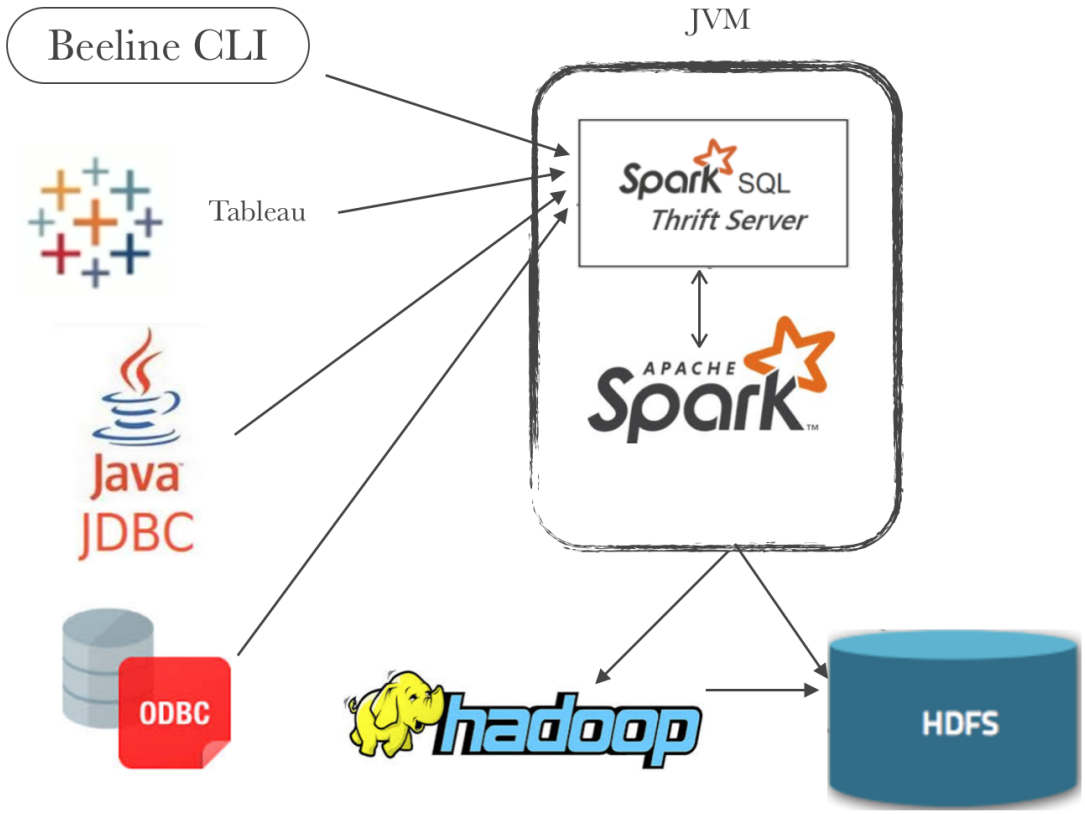
Spark applications run as independent sets of processes on a cluster, coordinated by the SparkContext object in a main program called the "driver" program. The Query Service has its own driver program and SparkContext instance, from which all the in-memory browse data can be accessed.

Spark can run standalone where all the necessary components are loaded at run time and jobs are executed. However, this method is (a) slow to instantiate, because of the need to load the components; and (b) difficult to manage, because each process has its own Java heap memory and resource requirements. In the Action Center context, the Query Service driver program connects to the Spark Cluster Manager, which accepts job requests and allocates resources across applications. Once connected, Spark acquires executors on nodes in the cluster, which are processes that run computations and store data for the application. Next, it sends the application code packaged in JAR or Python files to the executors. Finally, the SparkContext dispatches tasks to the executors to be run.



ThriftServer

The Query Service originally used Spark's native Java API to read and manipulate the browse data. With the introduction of Logi Platform Services in the Sep 2019 release, the data residing in the Query Service is retrieved directly by Logi as an SQL data source. This is accomplished by a component of Spark called the ThriftServer.

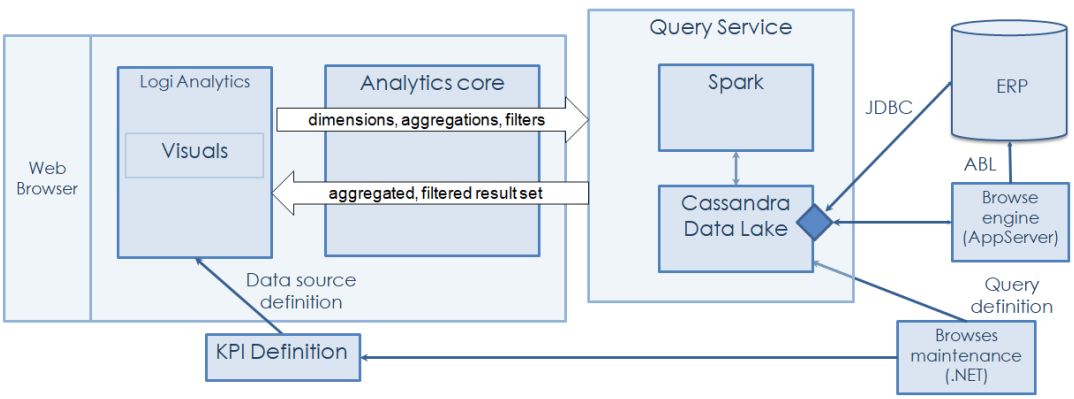


The ThriftServer is a server interface within Spark that enables remote clients to execute SQL queries and retrieve the results through a JDBC connection. In essence, it exposes the tables and views within Spark as an SQL database, supporting multi-client concurrency and authentication.

The ThriftServer is embedded inside the Query Service, which runs inside Tomcat. It requires the configuration of an additional port for Logi to connect through JDBC, but does not run inside a separate process.

Spark and Cassandra in the Query Service

In the Query Service, Spark and Cassandra are integrated to use the strengths and features of both to bring browse data to the Action Centers in a flexible and performant manner.



Spark serves as the in-memory cache where the data retrieved from browses is stored for on-demand retrieval. Before returning the browse results, it groups the data, pre-aggregates the numeric values within each group, and applies filters in order to return the minimum amount of detail needed to support Action Center display. Because grouping, aggregation, and filtering requirements can be changed by Action Center users at any time, Spark's combination of flexibility and speed is critical.

Cassandra serves as the data lake where the browse data is persisted, generally before it is needed. On a scheduled basis, browses that are configured for use in the Action Centers are refreshed from the operational OpenEdge database tables through one of two query mechanisms.

1. SQL with JDBC connections: Business Component browses are processed as SQL queries directly against the OpenEdge database. This approach is preferred, as it is significantly faster than the AppServer-based approach.
2. AppServer-based browse engine: Other browses are processed using the existing browse engine, which runs inside Progress AppServer agents and reads the OpenEdge data using ABL code.

End users can also request refreshes of a specific browse from the Action Centers, which causes the data to be refreshed in both Cassandra and Spark. In addition, beginning with the Sep 2020 release, historical snapshots of some KPIs are automatically created, stored in Cassandra, and cached in Spark. In future releases, browse results may be pushed into Cassandra more continuously as the source data is updated in Enterprise Applications through transaction processing activity.

The current KPI browse results extracted from the OpenEdge databases are stored in tables that reside in the Cassandra keyspace "browses." For most browses, there is a single table for each KPI and combination of browse and domain or browse and entity, depending on whether the browse was defined to access financial data, which are generally associated with financial entities, or operational data, which are generally associated with domains. The contents of these tables are then cached in Spark for online retrieval by the Action Centers.

This historical KPI snapshots are stored in table that reside in the Cassandra keyspace "historical_kpi". In this keyspace, there is a single table for each KPI snapshot. Because the number of historical snapshots per KPI can vary widely depending on KPI configuration, different historical KPIs can have different number of snapshot tables in the keyspace.

Action Center Installation

Review this section carefully before running the YAB installation, as it covers YAB properties that must be set and several procedures that must be completed prior to installation.

The Installation section describes useful details about how Action Centers and the related Logi and Query Service infrastructures are installed. It is not a comprehensive, step-by-step guide, as the installation process is largely automated through the use of the YAB tool. Action Centers are not installed in isolation, but as part of an overall release as documented in the *Adaptive UX On-Premise Installation Guide*. However, this section describes some installation steps in greater depth that are specific to Action Centers, referring to YAB command details and other guides as needed for context. It also covers important steps that must be completed before the automated installation is run.

This document assumes that the reader is familiar with basic Linux system administration and YAB. For more details on YAB, see the latest *QAD Configuration and Administration Guide* for YAB, available on the QAD Document Library.

System Requirements

Memory

Spark and Cassandra are fast because they do considerable amounts of in-memory processing. Logi Platform Services and Logi Info also require memory to render the visuals. The minimum memory requirement for running production systems with the Action Centers and Query Service is 16GB.

CPU

Systems running the Action Centers and Query Service should have a minimum of four cores.

Software

Java 8 and Python 2.7 in particular are required to run Cassandra and Spark. See the *QAD Adaptive UX On-Premise Installation Guide* for other software prerequisites.

Analytics (PLA) License

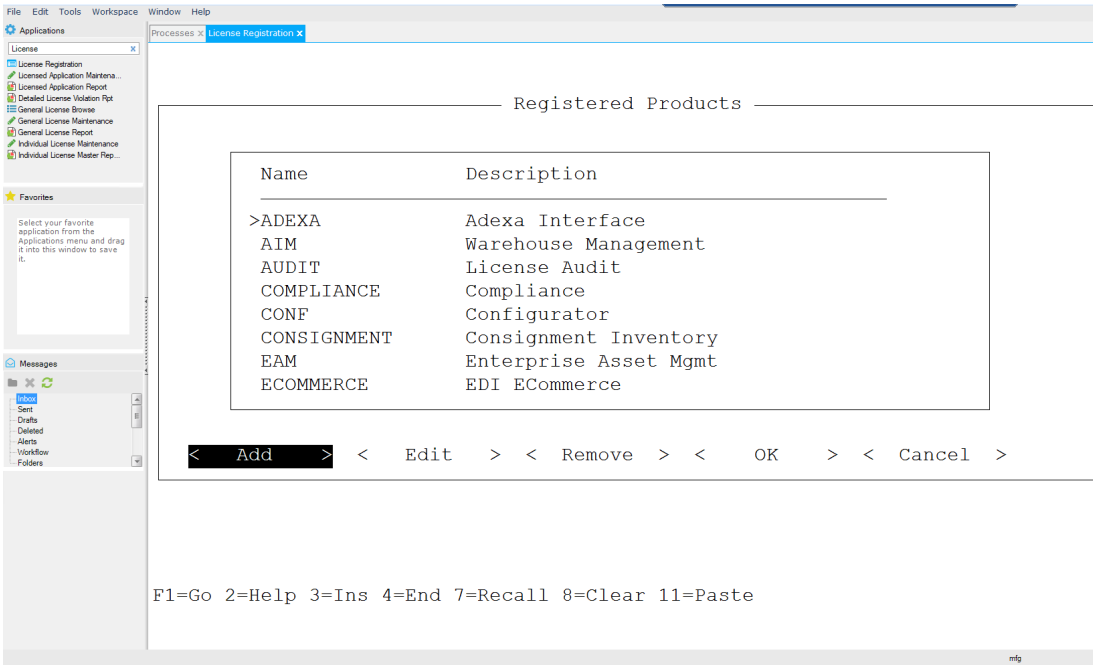
Starting with the September 2019 release, all QAD users who use Action Centers and KPIs must be registered to use the QAD Platform Analytics (PLA) license. The PLA license is provided as part of Enterprise Edition and administered in the same way as other QAD Enterprise Edition licenses. Users can be registered to use the license only with the License Registration screen in the QAD .NET UI. Licenses cannot be queried or updated in the Web UI. To avoid errors during the QAD Adaptive ERP installation, the PLA license should be present in the target environment and registered to appropriate users before installing or upgrading Adaptive ERP using YAB.

If the batch user is not registered for the PLA license using this procedure, the YAB installation/update will fail and corrective action will be needed as described later in this section.

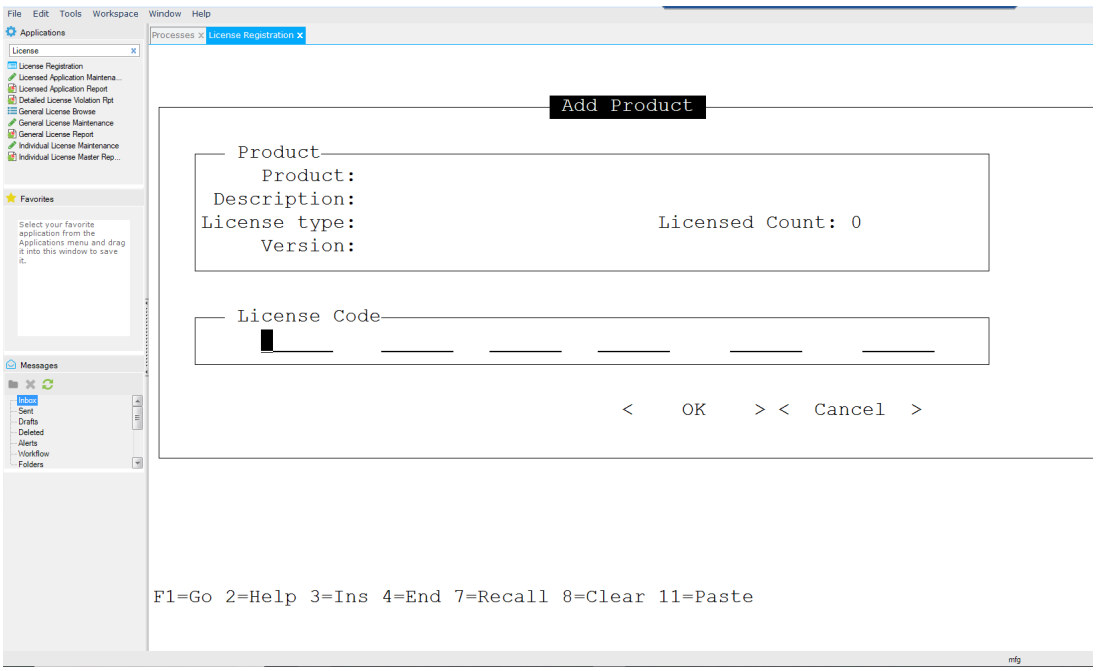
Adding PLA license and license codes

When installing or upgrading Adaptive ERP on Enterprise Edition installations prior to 2019, do the following before running the Adaptive ERP installation or upgrade.

- Log into the .NET UI as an admin user, search for the License Registration screen, and open it.
- Scroll down the list of licenses, checking for the PLA license.
- If the PLA license is already present in the list, skip to the procedure 'Registering Users for PLA License' below.
- If the PLA license is not present in the list, obtain the correct PLA license codes to use for the environment. Customer codes are obtained from QAD Global Customer Administration (GCA).
- Navigate to the Add button and press Enter.



- Enter the license codes for the PLA license using the .NET UI License Registration screen.

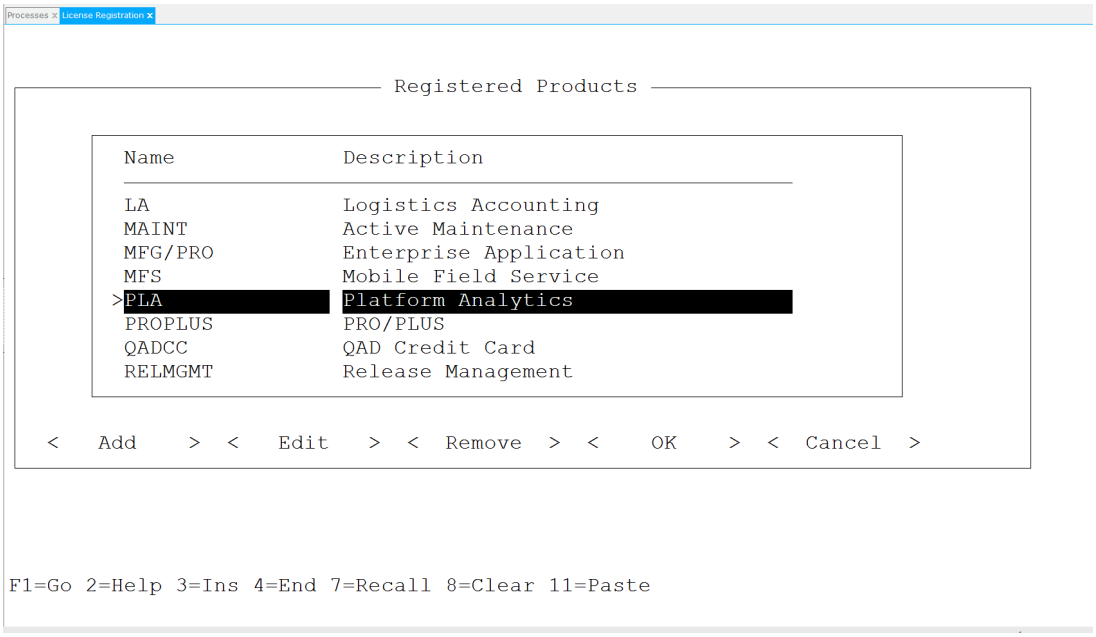


- Navigate to the OK button and press Enter.
- Confirm the changes when prompted by the UI.

Registering Users for PLA License

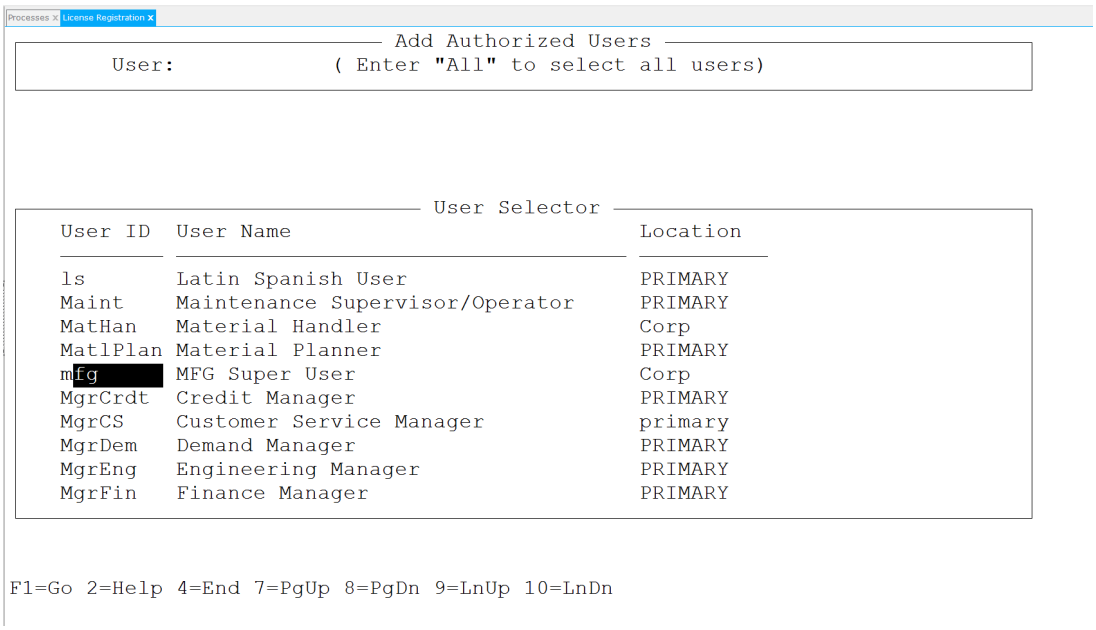
For all installations of and upgrades to the September 2019 or later release, do the following before running the Adaptive ERP installation or upgrade.

- Log into the .NET UI as an admin user, search for the License Registration screen, and open it.
- Scroll down the list of licenses, checking for the PLA license.
- If the PLA license is not present in the list, follow the previous procedure, 'Adding PLA License and License Codes.'
- Select the PLA license from the list.

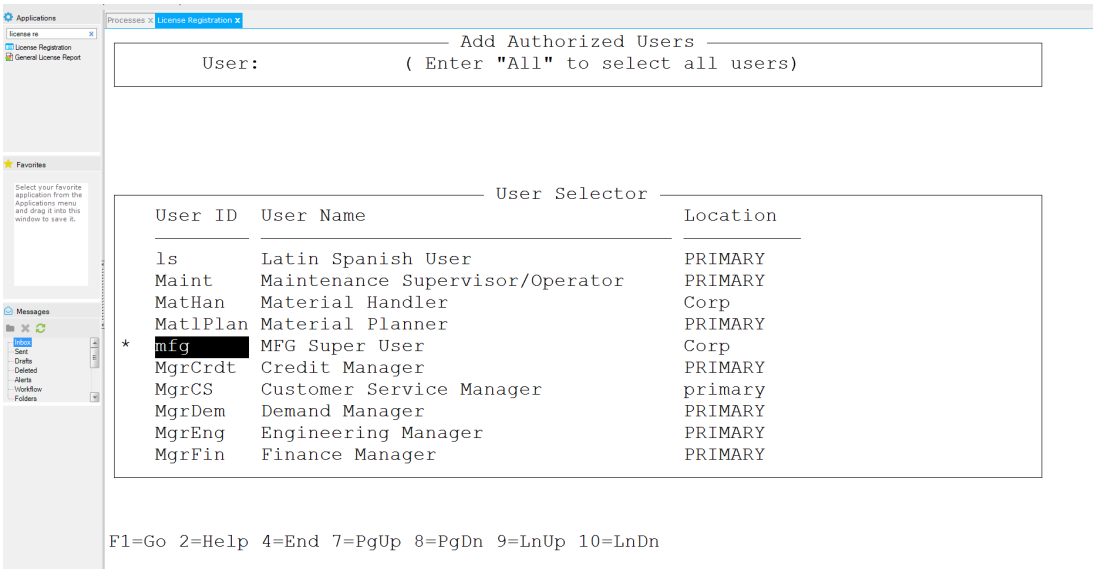


- Navigate to the OK button, and press Enter to access the list of registered users.

- To register a single user, in particular the 'batch user,' to use the PLA license, press Enter to see the list of users.



- Navigate to the user to be registered, and press Space to select the user. An asterisk will be displayed to the left of the user.



- Press the GO key to process the change, and confirm it when prompted by the UI.
- Alternatively, to register all users in the environment to use the PLA license, enter "All," press the GO key (usually F1), and confirm the action when prompted by the UI.

Error Caused by Missing PLA License Registration for Batch User

If the preceding step to register the QAD batch user for the PLA license is omitted, the YAB installation will fail with errors in an **lps-*-referencedataview-update** step written to the YAB log as shown below.

```
DEBUG [main] HttpCallCommand - POST https://vmltdc0078.qad.com:22011/qad-central/api/analytics/lps
/deployment/artifacts?type=referenceDataview&appUri=urn:app:com.qad.assetmgmt HTTP/1.1
DEBUG [main] HttpCallCommand - HttpResponseProxy{HTTP/1.1 403 Forbidden [Server: Apache-Coyote/1.
1, Set-Cookie: JSESSIONID=6BDC450A1A4CFB07E6418836F93F494C; Path=/qad-central; Secure; HttpOnly,
Cache-Control: no-store, X-Frame-Options: SAMEORIGIN, Content-Type: application/json;charset=UTF-
8, Transfer-Encoding: chunked, Date: Wed, 14 Aug 2019 21:36:30 GMT] ResponseEntityProxy{[Content-
Type: application/json;charset=UTF-8,Chunked: true]}}
DEBUG [main] APPLY - lps-assetmgmt-referencedataview-update ERROR
ERROR [main] Main - BUILD FAILED (1:12 h)
java.lang.RuntimeException: HTTP/1.1 403 Forbidden
        at com.qad.build.java.tasks.http.HttpCallCommand.invokeToResponse(HttpCallCommand.java:
181)
        at com.qad.build.java.tasks.http.HttpCallCommand.invoke(HttpCallCommand.java:47)
        at com.qad.yab.qra.QraWebuiApiClient.submit(QraWebuiApiClient.java:115)
        at com.qad.yab.qra.QraLpsArtifactUpdateProcess.execute(QraLpsArtifactUpdateProcess.java:
171)
        ...
```

The previous example references the 'assetmgmt' app, but the specific app raising the error in other environments could be different.

This error is triggered by missing permissions for the batch user, caused by the fact that the user has not been registered for the PLA license. To correct the problem, do the following.

- Complete the steps in the section 'Adding PLA license and license codes,' if the PLA license and codes do not already exist in the environment.
- Register the batch user for the PLA license as described in the section 'Registering Users for PLA license.'
- Run the following YAB command to re-execute the failed YAB command(s) that failed in the first attempt.

```
yab -clean lps-artifact-update
```

- Re-run the YAB installation or update.

Logi Licenses

Logi Platform Services

Logi Platform Services is required to support Action Centers created with September 2019 and later releases. It is a proprietary product that requires a license file obtained from QAD.

To use Action Centers beginning with the March 2020 release, a Logi license is assigned to the target machine and installed automatically during the installation for the production environments of most on-premise customers. In this case, no Logi license file is provided separately by QAD. However, automatic license assignment may not work for customers with firewall restrictions or internet connectivity problems on the target machine. If the Logi Platform Services license cannot be assigned automatically, a trial license with a 15-day expiration period will be installed so that Action Centers can be used for a short period of time until a permanent license is obtained. To obtain and install a permanent license in this case, several manual steps are required.

1. Request and obtain a Logi license file from QAD Fulfillment.
2. Follow the procedure described for the September 2019 release to install the license file.

In the case of non-production (development and test) environments using the March 2020 or later release, a separate Logi license is installed automatically by YAB with no need to obtain a machine-specific license from Logi or from QAD Fulfillment. It is important that non-production environments are not assigned production licenses from Logi, as production licenses incur a higher cost.

To use Action Centers in the September 2019 release, you must apply the Logi license file you received from QAD Fulfillment. Complete the following steps.

1. Save the Logi Platform Services license file contents in the file system with the name **lgx030505.lic**. Make sure that the name used for the Logi Info license file is not assigned by mistake.
2. Set the YAB properties **logi-platform-services._base.licensefile** and **logi-platform-services.default.licensefile** to the full pathname of this file (example: logi-platform-services.default.licensefile=/dr01/qadapps/temp/lgx030505.lic).
3. Run the YAB installation or update command.

Since the original March 2020 and September 2020 releases of Adaptive UX, several problems in the LogiPS licensing process described previously have been fixed. In order to avoid problems during installation related to Logi licensing, QAD strongly recommends that the latest available patch versions for these Adaptive UX releases be included in the installation.

Logi Info

Logi Info is required to support Action Centers created in releases before the September 2019 release. Logi Info is a proprietary product that requires a license file obtained from QAD. To use Action Centers created before the September 2019 release, you must apply the Logi license file you received from QAD.

The Logi Info license file contents should be saved in the file system with the name **lgx120102.lic**. Make sure that the name used for the Logi Platform Services license file is not assigned by mistake. This file should be copied into the root directory of the **qad-dashboards** webapp (example: /dr01/qadapps/systest/servers/tomcat-webui/webapps/qad-dashboards) before running the YAB installation or update. To find this root directory run the following command.

```
yab config webapp.analytics-logi.dir
```

Logi Platform Services Upgrades in the September 2019 Release

When upgrading an existing QAD Adaptive UX September 2019 environment to a service pack that includes a newer version of Logi Platform Services, several additional steps are required before running the full YAB update.

This section applies only to existing September 2019 installations with Logi Platform Services that are upgrading to a newer Logi Platform Services service pack. It does not apply to installations that did not previously contain Logi Platform Services, such as new installations or upgrades of environments that are older than September 2019. It also does not apply to QAD Adaptive UX installations of the March 2020 release or later.

Stop the Environment

First, stop the Adaptive ERP environment. Afterward, make sure that Logi Platform Services is no longer running.

```
yab stop
yab logi-platform-services-status
```

Back Up Logi Platform Services Database

If the environment contains Action Centers and visuals that have not been exported and released as part of an app, manually copy the Logi Platform Services database to a temporary file system location so its contents will not be lost during the upgrade. The Logi Platform Services database is stored in the location **<Adaptive ERP root>/servers/logi-platform-services/default/platform/db/**.

```
mkdir /mydirectory/temp
cp <Adaptive ERP root>/servers/logi-platform-services/default/platform/db/LogiDB.mv.db
/mydirectory/temp/
```

Upgrade to the new Service Pack

Run the following YAB command, which should take only a moment to complete.

```
yab logi-platform-services-default-rebuild
```

Restore Logi Platform Services Database

After the rebuild has completed, manually restore the Logi Platform Services database backup to its original location, if a backup was taken.

```
cp /mydirectory/temp/LogiDB.mv.db <Adaptive ERP root>/servers/logi-platform-services/default
/platform/db/
```

Run YAB Update

After the previous steps are complete, run the normal YAB update.

```
yab update
```

HTTPS Certificates for Logi Platform Services

This document assumes that Logi Platform Services will always be accessed using HTTPS, which is secure HTTP. If the Web UI is set up to be accessed through HTTPS, then Logi Platform Services must be configured to use HTTPS also, to avoid network security errors raised at run time. By default, Logi Platform Services is installed in QAD Adaptive ERP to be accessed through HTTPS, not basic HTTP.

Creating HTTPS Certificates

To use HTTPS with Logi Platform Services, a public certificate file and a private key file are required. However, the certificate and key files in the Java Keystore (JKS) format used by Tomcat do not work with Logi Platform Services. You must either generate new files in the PEM format, a common standard for storing cryptographic data, or convert existing JKS certificate and key files to that format.

Certificate and key files are not included in the Adaptive ERP installation and are not generated by YAB, but must be created on-site specifically for each company that is installing the software. They can be created in different ways, and there is no prescribed procedure. However, below is a simple procedure that has been used to create self-signed certificates for Logi Platform Services with the help of the open source tool openssl. This procedure converts an existing JKS keystore file into certificate and key files that can be used by Logi Platform Services. In the example, the environment variable JAVA_HOME must be set to the location of a JDK 8 installation on the local server.

```
$JAVA_HOME/bin/keytool -importkeystore -srckeystore myKeystore.jks -destkeystore myKeystore.p12 -
deststoretype PKCS12  # Convert JKS to PKCS 12 syntax
openssl pkcs12 -in myKeystore.p12 -nokeys -out myKeystore.crt  # Extract certificate file in PEM
format
```

```
openssl pkcs12 -in myKeystore.p12 -nocerts -nodes -out myKeystore.key  # Extract key file in PEM
format
```

The previous steps are sufficient to create a self-signed certificate. However, to generate a certificate signed by a Certificate Authority, the correct root certificate must also be downloaded from the Certificate Authority and appended to the *.crt file generated above. Below is a sample procedure to do this, once the root certificate ('myCA.crt' in the example) has been downloaded.

```
openssl x509 -inform der -in myCA.crt -out myCA.pem  # Convert the root certificate from binary
(DER) to textual X.509 (PEM) format
cat myCA.pem >> myKeystore.crt  # Append the root certificate to the certificate file
```

Configuring Certificate Files and Location

Once the certificate files are available, they must be placed in a fixed location on the file system where they can be read by YAB and copied into the Logi Platform Services installation. Various YAB properties must be set describing the files so that Logi Platform Services can consume them. The required properties are described in the following table.

Property Name	Description	Default Value
logi-platform-services.default.sourcekeyfile	Full path of the filename containing the private key to be used by LogiPS.	N/A
logi-platform-services.default.sourcecertfile	Full path of the filename containing the public certificate to be used by LogiPS.	N/A
logi-platform-services.default.selfsigned	Indicates if the certificate is self-signed.	false

As with other YAB properties, these should be set in configuration.properties.

Apache Reverse Proxy Configuration for Logi Platform Services

Many Adaptive ERP installations, including those hosted in the QAD Cloud, deploy an Apache web server as a reverse proxy in front of the Tomcat container to intercept and forward all web requests through a single port exposed to the internet. This kind of reverse proxy can enhance network security generally, which is not a topic covered in this document. However, in environments where an Apache reverse proxy is being used, a particular configuration change is required to support Logi Platform Services correctly.

In Adaptive ERP installations, the Apache configuration files contain a Location element that references the URL path of the reverse proxy, in this case 'clouderp.' Below is a sample from one such environment.

```
<Location /clouderp>
   ProxyPass https://vmlqad0000.qad.com:22011/qad-central
   ProxyPassReverse https://vmlqad0000.qad.com:22011/qad-central
   Header edit Set-Cookie "^(.*)/qad-central(.*)$" $1/clouderp$2
   Header edit Location "/qad-central/" "https://vmlqad0000.qad.com/clouderp/"
   AddOutputFilterByType SUBSTITUTE text/html image/svg+xml
   Substitute "s|/qad-central|/clouderp|iq"
   Substitute "s|https://vmlqad0000.qad.com:22011|https://vmlqad0000.qad.com|iq"
</Location>
```

In order to support Logi Platform Services, change the AddOutputFilterByType element in the previous example to the following, adding the MIME type 'application/com.qad.webshell.proxy+json' to the end.

```
AddOutputFilterByType SUBSTITUTE text/html image/svg+xml application/com.qad.webshell.proxy+json
```

Once this change is made, restart the Apache server. This is done outside of YAB, with no need to restart Tomcat or any other component of Adaptive ERP.

Installing Action Centers

Many QAD application packages include pre-defined Action Centers and KPIs. These standard Action Centers can be used as samples and starting points for creating custom Action Centers to suit the needs of different parts of the organization. As of the September 2019 release, the following packages include Action Centers built using Logi Platform Services.

- Assetmgmt
- Financials
- Fixedassets
- Inventory
- Planning
- Purchasing
- Pushproduction
- Sales
- Service

In later releases, Action Centers and KPIs were added for the Custrelmgmt (CRM) package.

In earlier releases, the QAD-provided Action Centers were built using Logi Info.

Action Center Installation Commands

In terms of technical deployment, Action Centers and their related data fall into three parts, all of which are included in the standard Adaptive ERP installation.

- Metadata loaded into the QAD database
- JSON objects loaded into the Logi Platform Services environment (for Action Centers created in the September 2019 release or later)
- XML data files copied into the Logi Info environment (for Action Centers created before the September 2019 release)

The Action Center metadata for the QAD database is loaded for each app as part of the YAB *metadata-update* command and its sub-commands. To install the metadata for a single package, including its Action Centers, run the YAB command with the syntax *metadata-<package>-update*. For example, the Sales Action Centers are installed by the YAB command *metadata-sales-update*.

The Action Center objects for Logi Platform Services are loaded for each app by a set of YAB commands, each of which loads a different kind of Logi object.

- lps-<package>-tag-update
- lps-<package>-referencedataview-update
- lps-<package>-enrichmentdataview-update
- lps-<package>-visualization-update
- lps-<package>-crosstabtable-update
- lps-<package>-dashboard-update

For each app, these commands must be run in the order shown. The command *lps-artifact-update* loads all of them in the prescribed order for all apps in the environment.

The Action Center files for Logi Info are copied for all apps at the same time for each file type (dashboard, KPI, gallery) by the following YAB commands.

- action-center-dashboard-update
- action-center-kpi-update
- action-center-gallery-update

Action Centers Developed Using Logi Info are Installed But Not Updated or Deleted

In releases prior to September 2019, there is an important difference in the handling of Action Centers vs other kinds of metadata. These Action Centers, built using the older Logi Info framework, are installed by YAB, but not updated or deleted once they are installed. Standard Action Centers may be used off the shelf and modified after installation to meet local requirements. If the standard Action Centers were routinely updated as part of a system or package upgrade, local modifications would be overwritten and lost. While any application packages containing Action Centers can be upgraded, data related to existing Action Centers and KPIs is skipped during the YAB update.

To update Action Centers and KPIs inside one environment with Action Centers and KPIs of the same name that have been created in a different environment (for example, migrate updated versions of an Action Center and its associated KPIs from a development environment to production), use the KPI Migrate function in the Web UI to export the KPIs from the source environment and import them into the target environment. There is no similar function to export and import an Action Center, so a modified Action Center must be updated manually in the target environment. However, updating an Action Center using Logi Info consists mainly of removing/adding/rearranging dashboard panels, and this is usually a quick process.

When updating apps that were developed outside the organization (for example, a recent release of a previously installed QAD package), newer versions of Action Centers and KPIs that already exist in the target environment are not updated for the reasons previously mentioned. In this case, the source environment in which the Action Centers and KPIs were

defined is not available and the KPI Migrate function cannot be used to export and import them. If the newer versions of these predefined Action Centers and KPIs are needed, please contact QAD Support for assistance.

When updating apps that include Action Centers and KPIs new to the target environment, no special steps are needed. The new Action Centers and KPIs are installed automatically as part of the YAB update.

Action Centers Developed Using Logi Platform Services are Installed, Updated, and Deleted

As of the September 2019 release, the preceding points do not apply. Action Centers and KPIs are developed using Logi Platform Services, not Logi Info. In addition, Action Centers and KPIs that are developed in one app (for example, one of the standard QAD apps previously listed) cannot be updated or deleted by users whose active app is different. In order to use a standard QAD Action Center as the starting point for a custom version, a user can clone the Action Center and its associated KPIs into the active app using Save As and Copy functions in the Web UI. The cloned versions can then be modified as desired, eliminating the need to prevent standard QAD Action Centers from being updated. Therefore, Action Centers and KPIs beginning with the September 2019 release built using Logi Platform Services are add-updated-deleted during YAB updates in the same way as other application artifacts.

Exporting Action Centers

The App Name field shows the app in which the KPI data is stored. The field is read-only, and is set to the app selected as your active app in the My Developer Settings screen. Action Centers and KPIs are automatically exported with that app using the YAB *app-export* command. The output of the export written to the specified directory includes the following files related to Action Centers. These files are generated by the Action Center infrastructure and should not be manually edited.

When you migrate a KPI from a source environment to a target environment, the migration does not include the browse that is used as the data source. You must ensure the browse used as the data source already exists in the target environment and that it has the same fields in its definitions. You can migrate browse definitions with the import/export tool in Browse Maintenance in the QAD .NET UI.

Action Centers Created In September 2019 and Later Releases (Logi Platform Services)

- data/analytics/ sub-directory
  - **D-*.xml** files: Database records defining an Action Center. These records allow permissions to access particular Action Centers to be granted or revoked based on role.
  - **K-*.xml** files: Database records defining a KPI. These records define the data fields, filters, and domains /entities significant for a particular KPI.
- ac/lps/ sub-directory
  - **dashboard/**: Directory of LogiPS database contents for Action Centers in JSON files.
  - **enrichmentdataview/**: Directory of LogiPS database contents for enrichment dataviews in JSON files.
  - **referencedataview/**: Directory of LogiPS database contents for reference dataviews in JSON files.
  - **tag/**: Directory of LogiPS database contents for tags in JSON files.
  - **visualization/**: Directory of LogiPS database contents for visuals in JSON files.

Action Centers Created Before September 2019 Release (Logi Info)

- data/analytics/ sub-directory
  - **D-*.xml** files: Database records defining an Action Center. These records allow permissions to access particular Action Centers to be granted or revoked based on role.
  - **K-*.xml** files: Database records defining a KPI. These records define the data fields, filters, and domains /entities significant for a particular KPI.
- ac/dashboard/ sub-directory
  - **Dashboard-*.xml** files: Logi Info file defining the layout and contents of an Action Center.
- ac/gallery/ sub-directory
  - **Gallery.xml** file: Logi Info file containing information about the visuals in the exported package published for use in Action Centers.
- ac/kpi/ sub-directory
  - **AGState-*.xml** files: Logi Info file defining the layout and content of the visuals displayed from an expanded Action Center panel, or by pressing the Visuals button for a particular KPI in the KPI screen.

YAB Commands

Logi Platform Services

To obtain a list and description of all YAB commands that can be used to monitor and control Logi Platform Services, run the following command.

```
yab help logi-platform-services-
```

To obtain a list and description of all YAB commands that are used to install and extract Action Centers and related objects to/from Logi Platform Services, run the following command. The YAB commands described are run by YAB during Action Center deployment.

```
yab help lps-
```

Most of the Logi Platform Services commands documented in the YAB help are run only by YAB, and would not normally be run directly from the command line. However, some of them can be useful to system administrators. The following YAB commands would normally be run from the command line for routine monitoring and control purposes. Any of them can be run without the need to stop/start Tomcat or other Adaptive ERP components.

YAB Command	Description	Remarks
logi-platform-services-default-application-start	Starts the application service for the Logi Platform Services 'default' instance.	The data service must be started before the application service is started.
logi-platform-services-default-application-status	Returns status of the application service for the Logi Platform Services 'default' instance.	
logi-platform-services-default-application-stop	Stops the application service for the Logi Platform Services 'default' instance.	The application service must be stopped before the data service is stopped.
logi-platform-services-default-data-start	Starts the data service for the Logi Platform Services 'default' instance.	The data service must be started before the application service is started.
logi-platform-services-default-data-status	Returns status of the data service for the Logi Platform Services 'default' instance.	
logi-platform-services-default-data-stop	Stops the data service for the Logi Platform Services 'default' instance.	The application service must be stopped before the data service is stopped.
logi-platform-services-default-start	Starts the 'default' Logi Platform Services instance.	Combines the logi-platform-services-default-data-start and logi-platform-services-default-application-start commands.
logi-platform-services-default-status	Returns the status of the 'default' Logi Platform Services instance.	
logi-platform-services-default-stop	Stops the 'default' Logi Platform Services instance.	Combines the logi-platform-services-default-application-stop and logi-platform-services-default-data-stop commands.
logi-platform-services-default-restart	Stops and immediately re-starts the 'default' Logi Platform Services instance.	Tomcat-webui does not have to be restarted when LogiPS is restarted.
logi-platform-services-license-info	List the LogiPS license(s) installed in the Logi Platform Services 'default' instance.	Useful for checking the presence or type of a LogiPS license in case of license-related errors.
logi-platform-services-license-	Assigns or re-assigns and installs a LogiPS license to the Logi Platform Services	The type of license assigned depends on whether the environment type is development, test, or production.

assign	'default' instance.	
logi-platform-services-license-unassign	Un-assigns and deletes the currently assigned LogiPS licenses from the Logi Platform Services 'default'instance.	May be needed if the wrong kind of Logi license was installed for any reason, or if the production environment is being deleted or its server de-commissioned. In the latter case, failure to un-assign the license would trigger the purchase of an extra LogiPS production license unnecessarily.

Logi Info

To obtain a list and description of all YAB commands that can be used to deploy and extract Action Center files used by Logi Info, run the following command, which gives details for the three types of Action Center files: dashboard files, gallery, and KPI files.

```
yab help action-center-
```

Cassandra

To obtain a list and description of all YAB commands that can be used to administer Cassandra, run the following command.

```
yab help cassandra-
```

For a list of all Cassandra-related settings, including the YAB commands, run the following command.

```
yab help cassandra
```

Following are the commands that would most commonly be run from the command line.

YAB Command	Description	Remarks
cassandra-default-status	Checks the status of the 'default' Cassandra instance.	
cassandra-default-start	Starts the 'default' Cassandra instance.	
cassandra-default-stop	Stops the 'default' Cassandra instance.	
cassandra-default-restart	Stops and re-starts the 'default' Cassandra instance.	Spark and tomcat-webui do not have to be restarted when Cassandra is restarted.
cassandra-default-nodetool	Runs the Cassandra nodetool command-line utility.	Run 'yab cassandra-default-nodetool -command:help' for a list of subcommands in the nodetool utility, and 'yab cassandra-default-nodetool -command:<subcommand> to run any of them. For more detailed background about each one, see the Cassandra web pages.

Spark

To obtain a list and description of all YAB commands used to administer Spark, run the following command.

```
yab help spark-
```

For a list of all Spark-related settings, including the YAB commands, run the following command.

```
yab help spark
```

Most of the Spark commands documented in the YAB help will be rarely used, especially given that all AUX releases deploy Spark in a non-clustered manner on a single server. Following are the ones that would most commonly be run from the command line.

YAB Command	Description	Remarks
spark-status	Checks the status of all Spark nodes and processes.	
spark-start	Starts the Spark master and slave (worker) processes.	
spark-stop	Stops the Spark master and slave (worker) processes.	
spark-restart	Stops and re-starts the Spark master and slave (worker) processes.	If Spark is restarted, tomcat-webui must also be restarted in order to restore Query Service connections to Spark.

Action Center Security

This section describes various features, configuration settings, and considerations related to security in the Action Center and Query Service.

Action Centers

Roles and Permissions

Action Centers are part of the QAD Web UI and are displayed on the Web UI menu. The permissions required to access them in different ways are restricted by role using the same security infrastructure as other Web UI screens. For detailed information on setting permissions, see the *QAD Security Administration Guide,* available on the QAD Document Library.

Permissions are granted by role to create Action Centers, and to view and delete particular Action Centers. 'Sharing' permissions can also be granted by role that allow authorized users to add, replace, and delete visuals in the common gallery. In addition, the user who created a particular Action Center always has full access permissions to it, regardless of his/her role. In this respect, Action Centers are different from other secured resources in Adaptive ERP. For more information on Action Center permissions, see the online help for the QAD Web UI (https://documentlibrary.qad.com/help/webui/2019_1/en-US/index.html#page/QAD%2520Online%2520Help%2FActionCenterPermissions.html%23).

Domain-Entity-Site Membership

Another aspect of Action Center security is the membership of users within particular domains, financials entities, and sites. This type of security is part of the common Web UI security infrastructure, and is not covered in this document. However, domain, entity, and site membership is used by the Action Centers to automatically filter the data that end users can view. For example, two users have permissions to view a particular Action Center, but User 1 is a member of domain 10USA and User 2 is not. In this case, both users would see the same panels and visuals in the Action Center but the visuals shown to User 1 would include data from domain 10USA, whereas the visuals shown to User 2 would not.

Logi Platform Services Authentication

Logi Platform Services has its own security model that has been integrated with QAD Adaptive UX, with the goal of minimizing the amount of overlapping security data maintained across the two systems. It supports two authentication mechanisms used by the Action Centers functionality of QAD Adaptive UX: native user authentication and trusted access authentication.

Native User Authentication

LogiPS can maintain its own information and credentials for individual users, without the need for those users to be QAD users that are recognized in QAD Adaptive UX. In fact, in order to install LogiPS, there must be at least one 'admin' user with full permissions to access and modify any data maintained in the LogiPS database. This 'admin' user is used by QAD Adaptive UX to make background API calls into Logi Platform that are not requesting information on behalf of QAD on-line users – for example, by YAB at installation/update time in order to configure the LogiPS environment. However, because LogiPS has been embedded into QAD Adaptive UX all LogiPS end users must be QAD end users, with the QAD users, roles, and permissions controlling both QAD Adaptive UX and LogiPS usage. In general, Logi Platform Services and the Action Centers that it contains can be accessed only through the Web UI, not by logging into LogiPS directly. For this reason, no users except for the required 'admin' user are created or maintained natively in LogiPS. Instead, the trusted access authentication mechanism described later in this document is used to enable QAD users to access LogiPS in a seamless manner, subject to the role-based permissions and domain-entity-site membership maintained in QAD Adaptive UX.

The credentials for the LogiPS 'admin' user are set at installation time by YAB and can be changed after installation. Changing the default password is strongly encouraged in order to keep the system more secure. The username 'admin' should not be changed. The YAB properties with default values are as follows.

```
logi-platform-services.default.username=admin
logi-platform-services.default.password=password
```

The password can be changed using the following YAB command.

```
yab logi-platform-services-default-password-update
```

Whenever the password is changed, both tomcat-webui and Logi Platform Services must be restarted.

Trusted Access Authentication

QAD user IDs are not maintained separately in LogiPS. Instead a single sign-on mechanism is used, where the Web UI passes required user information to LogiPS and authenticates a QAD 'trusted user' on the fly whenever needed, passing a client secret to LogiPS specific to the installation in order to vouch for the user's authenticity. The Web UI also passes important information about the user's permissions to LogiPS, so that the appropriate security rules can be defined and enforced in LogiPS to prevent users from retrieving unauthorized business data from Adaptive Applications.

This trusted access authentication mechanism is automatically applied and requires no system administration effort. It requires only a client ID and client secret code pair maintained in both LogiPS and the Web UI, which is generated by YAB automatically at installation time. For reference, the YAB properties in which these values are stored are as follows.

```
logi-platform-services.default.clientid=analytics
logi-platform-services.default.clientsecret=eyJhbGciOiJBMTI4S1ciLCJlbmMiOiJBMTI4Q0JDLUhTMjU2In0.
Fr4TBVdbNNNBnwdB7IAPUuXoVO5K-pT7KDL8SlAFjtMolXShXjg6cg.Kr5vg5dre2iKkMjlZByZEg.
KO3UzPUve149bmPTglKSCn-yahkEEIq7x-xgL-
3JtmeFs2B3do5mnXg6SrZgB59f_wG_Gk010YD4elkgzHbwLbekW0glns09MRwyDoYo2Ck.L04Buq0IrsPOSenuPA-UCA
```

The client secret is generated at installation time based on an encryption algorithm, and is specific to each environment. It can also be re-generated if needed by running the following YAB command, although this should not be required.

```
logi-platform-services-default-clientsecret-update
```

Whenever the client secret is re-generated, both tomcat-webui and Logi Platform Services must be restarted.

Logi Info Authentication

Logi Info is deployed as a separate Tomcat web application, named 'qad-dashboards' by default, but is intended to be accessed only from within the Web UI. Users are not allowed to access Logi Info or display Action Centers without first logging into the Web UI. To ensure that all access to the Action Centers is restricted to Web UI sessions, Logi *SecureKey authentication* is enabled.

With SecureKey authentication enabled, every Web UI request to display Action Centers or other Logi Info views is preceded by a server-to-server HTTP 'handshake' call from the Web UI webapp to the Logi Info webapp requesting a valid SecureKey token. A token is then returned to the Web UI, allowing the Web UI to include Action Center displays through a subsequent request. This SecureKey authentication handshake is processed quickly and is invisible to end users. While it is enabled by default and should require no manual installation or configuration steps, the properties controlling the processing are summarized here.

SecureKey authentication is enabled in the file qad-analytics-core.properties by the following property.

```
qad-analytics-core.logiSecureKeyEnabled=true
```

It is also enabled in the Logi Info configuration file _Definitions/_Settings.lgx in the XML Security element.

```
<Security AuthenticationClientAddresses="0.0.0.0 255.255.255.255" AuthenticationSource="
SecureKey" RestartSession="True" SecurityEnabled="True"/>
```

These properties should not be changed.

Query Service Security

Spark ThriftServer Authentication

Starting in the September 2019 release, a new component of Spark called the ThriftServer has been introduced into the Query Service. The ThriftServer allows Logi Platform Services to call the Query Service directly to retrieve required KPI and browse data using SQL requests. The ThriftServer exposes a network port that can be accessed by Logi Platform Services through a JDBC connection. The port used by the ThriftServer is assigned by YAB at installation time, and stored in the following property.

```
spark-thriftserver.default.port
```

Because the ThriftServer port allows browse results from Adaptive Applications to be retrieved using remote SQL clients with a valid JDBC connection, it must be secured to prevent unauthorized access. Because only Logi Platform Services needs to connect to this port, QAD recommends that outside access to it be blocked by a network firewall. In addition, access to the port is secured by username-password credentials that can be set in the following YAB properties.

```
spark-thriftserver.default.username
spark-thriftserver.default.password
```

YAB assigns default values to these properties. Changing the default password is strongly encouraged in order to keep the system more secure. The password can be changed in a YAB update, following by a restart of tomcat-webui and Logi Platform Services.

Cassandra Authentication

By default, authentication to access the Cassandra data lake is disabled. Whether authentication is enabled or disabled has no direct effect on Action Center users. However, the lack of authentication exposes security holes regarding access to the browse data that is stored in Cassandra. In particular, a user with the ability to run the Cassandra shell *cqlsh*, described in the Action Center Maintenance and Troubleshooting section, could connect to the data lake and view/update /delete the browse data without restriction using SQL commands. For this reason, it is strongly recommended to enable Cassandra authentication using YAB, so that valid username-password credentials are required to access any of its data.

To enable Cassandra authentication, set the following YAB property.

```
cassandra.default.node.main.authenticator=PasswordAuthenticator
```

Once Cassandra authentication is enabled, the Cassandra user and password are set to 'qad' and 'qad' respectively by default. Change these to more secure values for the installation by setting the following YAB properties.

```
cassandra._base.user
cassandra._base.password
```

Run a YAB update to process these changes.

Spark Web UI Access

Spark provides a native web user interface, completely separate from Action Centers, that can be used by system administrators to monitor the jobs and tasks launched and executed internally by Spark. This UI is described briefly in the Action Center Maintenance and Troubleshooting section of this document. It is a single set of integrated web pages, but the pages are actually made up of three UIs that can be accessed through separate network ports. While potentially useful, several of these UIs expose security risks to the run-time environment. The risks and the means of reducing or eliminating them through configuration are described in the following sections.

Driver UI

The 'driver' is the Query Service itself running inside Tomcat, which creates a Spark application by communicating to the stand-alone Spark cluster manager, called the 'master.' This UI allows users to view all Spark environment settings, both those set by YAB and those internal to Spark, which include internal Spark passwords. It also displays 'kill' hyperlinks that allow users to stop in-process Spark jobs, which disrupt Action Center processing. If a Spark job is killed in this manner, the Tomcat instance hosting the Web UI has to be restarted to ensure that the Action Centers work properly.

The port used to access the driver UI is configured by the following YAB property.

```
qad-qracore.spark.ui.port
```

The driver UI is enabled by default. To disable it entirely, set the following YAB property.

```
qad-qracore.spark.ui.enabled=false
```

The ability to kill Spark jobs from the driver UI is controlled by the following YAB property, set to false by default.

```
qad-qracore.spark.ui.killEnabled=false
```

Master UI

The 'master' runs in a separate Spark JVM process with responsibility for dispatching and tracking the work across a cluster of workers, which in the current release is only a single worker node. Like the driver UI, it displays 'kill' hyperlinks that allow users to stop in-process Spark jobs, which disrupt Action Center processing.

The port used to access the master UI is configured by the following YAB property.

```
spark._master.env.webui.port
```

The master UI is automatically enabled and cannot be disabled. However, the ability to kill Spark jobs from the master UI is controlled by the following YAB property, set to false by default.

```
spark._master.properties.spark.ui.killEnabled=false
```

To prevent all use of the master UI, the network firewall must be configured to block access to its port.

Worker UI

The 'worker' runs in a separate spark JVM process and performs queries-processing based on requests from the master node. Unlike the other Spark UIs, the worker UI does not expose any sensitive data or affect Spark processing.

The port used to access the worker UI is configured by the following YAB property. It is enabled by default and cannot be disabled.

```
spark._slave.env.webui.port
```

To prevent all use of the worker UI, the network firewall must be configured to block access to its port.

Logi Platform Services Product Database Security

Unlike the older Logi Info framework, Logi Platform Services includes a separate 'product database' (PDB), rather than storing all information inside individual XML files saved in the file system. This product database contains internal representations of the Action Centers, visuals, and KPIs, in addition to system configuration data. It is relational, implemented using the H2 database engine (see https://www.h2database.com/). The Logi Data Service within LogiPS connects to the PDB through a single configurable port, secured with login credentials.

The PDB is not accessed outside of LogiPS. Neither YAB nor QAD Adaptive UX ever reads or writes its contents directly, and there would be no reason for its connection port to be exposed outside of the enterprise firewall. While this port can be configured by YAB, the database connection credentials are native to LogiPS and not maintained using YAB.

The port used by the PDB for client connections is stored in the following YAB property, dynamically assigned by YAB at installation time.

```
logi-platform-services.default.service.data.h2.port
```

The PDB connection credentials are automatically set by LogiPS at installation time and would normally never need to be changed. However, LogiPS provides a command line utility dbPassword.sh that can be used to change them if needed. The help text for using this utility is shown below.

```
dbPassword: Change platform MQ or DB password

Usage

-v, --verbose <number>    Provide extensive output (0-3)
--help                    Print usage instructions
-o, --offline             Invoke command in offline mode
-c, --MQ                  Change Message Queue password
-d, --DB                  Change Database password
```

The '-d' option above is used to change the PDB password. Logi Platform Services should be stopped before using the utility. After the utility is run, Logi Platform Services must be restarted.

Logi Platform Services Network Ports

Starting with the September 2019 release and the introduction of Logi Platform Services into the QAD Adaptive UX technology stack, various network ports required by LogiPS are automatically configured by YAB. These ports have

different purposes within the Logi architecture with different security considerations. They are summarized in the following table.

Purpose of Port	YAB Property	Security Approach	Remarks	Required Access to Port
HTTPS client access to LogiPS	logi-platform-services.default.service.application.webserver.sslport	Native Logi Platform ('native user') authentication and Web UI ('trusted user') authentication	Used for Logi Platform API access	tomcat-webui server of QAD Adaptive UX only
HTTP client access to LogiPS	logi-platform-services.default.service.application.webserver.port	Native Logi Platform ('native user') authentication and Web UI ('trusted user') authentication	Used for Logi Platform API access, but disabled by default in favor of HTTPS	tomcat-webui server of QAD Adaptive UX only
H2 (PDB) database access	logi-platform-services.default.service.data.h2.port	Internal username-password credentials, automatically configured by LogiPS and not managed by YAB	Internal to Logi Platform. See earlier section describing the PDB for more details.	Internal to LogiPS
Embedded LDAP access	logi-platform-services.default.service.data.ldap.port	Configurable username-password credentials, with password encrypted in file by LogiPS and not managed by YAB	Internal to Logi Platform, little need to change password	Internal to LogiPS
ActiveMQ access - Stomp	logi-platform-services.default.service.data.stomp.port	Internal username-password credentials, automatically configured by LogiPS and not managed by YAB	Internal to Logi Platform, little need to change password	Internal to LogiPS
ActiveMQ access - Openwire	logi-platform-services.default.service.data.openwire.port	No authentication is needed, as protocol is binary and communications are internal to ActiveMQ transport within LogiPS	Internal to Logi Platform, little need to change password	Internal to LogiPS

As shown in this table, all the ports are internal to Logi Platform Services except for the HTTPS and HTTP service ports, which are accessed from the tomcat-webui server. None of them require direct access by client browsers.

Action Center Configuration

The Configuration section describes important details about the configuration of Action Centers and the Query Service components that support them. Most configuration properties are maintained using YAB. General reference information on any of the properties can be found by running the command:

```
yab help <property name>
```

This guide includes details about only a portion of the properties used to configure Action Centers. For descriptions of most of them, run the command:

```
yab help qad-analytics-core
```

For descriptions of the properties related specifically to Financial Report Writer KPIs, run the command:

```
yab help qad-analytics-financials
```

Configuration related to Action Center security is described in Action Center Security.

KPI Caches

KPIs defined in the Web UI describe the data sets that populate the Action Centers. In order to achieve acceptable performance, they are cached in memory for on-demand display in the Action Centers.

There are several different data sources for KPIs, all of which retrieve data from the OpenEdge databases.

- Browses
- Financial Report Writer (FRW)

The infrastructure used to cache the data for these types of KPIs are different, but most of the caching configuration is defined using properties common to all KPI types.

Browse-Based KPIs

KPIs that have browses as their data sources make up the majority of KPIs, because browses use a powerful data retrieval mechanism and can be defined by knowledgeable end users. They have the capability of retrieving data from almost any database table in the system, including financial and operational data.

Pre-defined browse-based KPIs are packaged and installed inside various apps, generally the same apps as the Action Centers that use them.

The browse result sets used by the browse-based KPIs are cached by the Query Service, using its Spark and Cassandra infrastructure.

In Cassandra, browse result sets are persisted to disk in tables that are specific to a combination of browse and entity for financial browses, and browse and domain for all other browses. These tables reside within the 'browses' keyspace. Following are some examples of table names displayed from this Cassandra keyspace.

```
cqlsh> use browses;
cqlsh:browses> describe tables;

kpi_357844472_12mex      kpi_1688405265_20fra     kpi__1355073026_10usa
kpi__846170097_12mex     kpi_1852174264_20fra     kpi__1863008167_22uk
kpi_1988561950_21nl      kpi_1988561950_12mex     kpi__692383936
kpi__1809347128_20fra    kpi__45865388_23ger      kpi__1451599441_10usa
kpi_1202448307_12mex     kpi_1044215493_21nl      kpi_1533800352_30chn
kpi_1202448307_30chn     kpi__1063222905_12mex    kpi_1384866991_10usa
kpi__186238821_12mex     kpi_1705405443_qad       kpi_1988561950_20fra
kpi_1650412229_11can     kpi__1798778746_20fra    kpi__1063222905_11can
kpi__1975819738_20fraco  kpi_1471838795_31aus     kpi_1105371684_10usa
kpi_1206509372_31aus     kpi_1384866991_11can     kpi__1423034166_21nl
kpi__1282164377_23ger    kpi__45865388_31aus      kpi__1809347128_12mex
kpi__1422684978_23ger    kpi__1319591205_22uk     kpi_642442275_10usaco
kpi__1942540096_12mex    kpi__1808255083_10usa    kpi__1604144893_31aus
kpi_1533800352_10usa     kpi__1423034166_11can    kpi__875669433
kpi__78780668_12mexco    kpi_1044215493_31aus     kpi_1529480452_31aus
kpi_1101437247_31aus     kpi_1175672099_12mex     kpi_1852174264_11can
```

```
kpi_1854608121_30chn     kpi_1471838795_21nl      kpi_1206509372_11can
kpi__1916630248_11can    kpi_1101437247_40brz     kpi_131336283_21nl
kpi_1080786763           kpi_131336283_22uk       kpi_220484148_12mex
kpi_514817284_10usa      kpi_1529480452_22uk      kpi__6799222_40brz
kpi_1817003066_10usa     kpi_1852174264_80tst     kpi_1533800352_20fra
kpi_1650823036_31aus     kpi__2068936752_30chn    kpi__1809347128_31aus
kpi__1798778746_10usa    kpi__2086541207_23ger    kpi_1721476749_23ger
kpi_1117887640_22uk      kpi_1854608121_40brz     kpi__778172591_10usa
kpi_514817284_11can      kpi_1721476749_40brz     kpi__1918291693_10usa
kpi__1143427273_12mex    kpi_1688405265_21nl      kpi__1537806489_40brz
...

cqlsh:browses>
```

The KPI tables whose names end in a QAD domain or financial entity code (example: "_10usa") contain data from that domain or entity only for a particular KPI. KPI tables with no domain suffix contain data from all domains or entities enabled for that KPI. Unfortunately, the KPI name and its source browse are not identifiable from the table name. Table names may be changed in a future Adaptive UX release to correct this.

The Cassandra tables are cached in memory by Spark, which creates views on the fly with appropriate filtering and grouping to support the needs of specific Action Centers.

Financial Report Writer KPIs

Beginning with the Sep 2019 release, KPIs based on the Financial Report Writer (FRW) tool have been deprecated in environments using Enterprise Financials 2019 and above. However, they are still needed in environments that use older Enterprise Financials versions, and may also be present in environments upgraded from earlier Adaptive UX releases.

The Financial Report Writer (FRW) is the primary financial reporting tool within the Financials application. It is maintained in the QAD .NET UI and has the capability to define financial KPIs based on the chart of accounts, organizational, and reporting structures specific to the enterprise. These FRW KPIs can organize and roll up financial measures in a manner consistent with the enterprise's financial reports, which could not be easily done through user-defined browses.

> ⚠ FRW-based KPIs can only show data in Action Centers if FRW reports have been set up in the QAD .NET UI and FRW-based KPIs are created in the QAD .NET UI with names that match the FRW-based KPI definitions. For information on how to set up FRW reports and create FRW-based KPIs in the QAD .NET UI, see *QAD Financials User Guide*.

FRW KPIs generally consist of smaller data sets than browse-based KPIs, and are not cached using the Query Service. Instead, they are cached in memory within the product's primary web application using the Java Ehcache library.

Entries in the FRW KPI cache are identified by a combination of KPI code and financial entity, as in the following examples.

KPI Code	Entity Code
Cash Flow Analysis	10USACO
Days Payable Outstanding	21NLCO
Net Profit Margin	22EMEACONS
Working Capital	30CHNCO

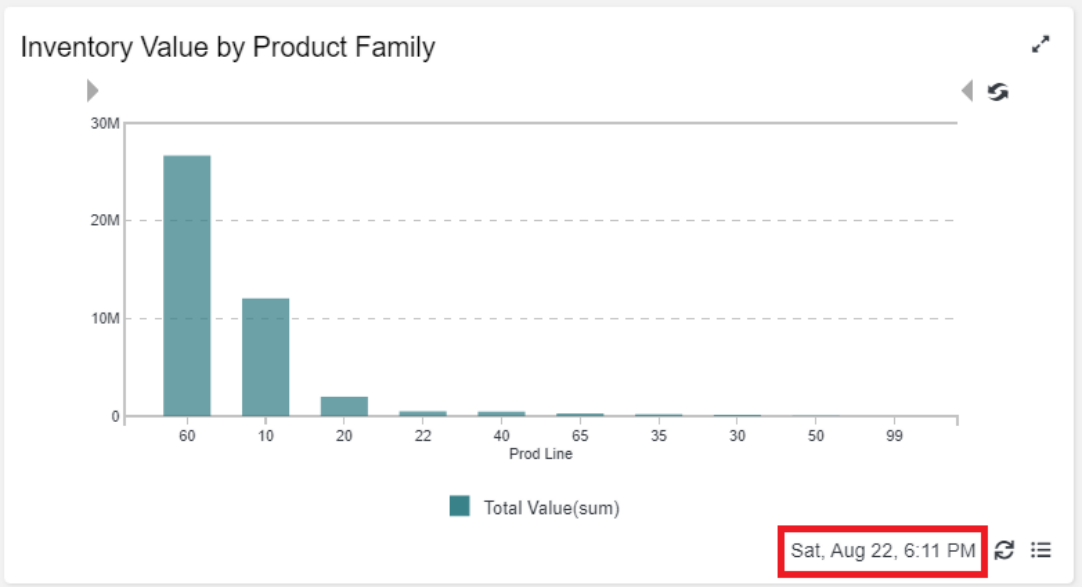
KPI Cache Refreshes

KPI caches can be refreshed with current source data using several configurable approaches.

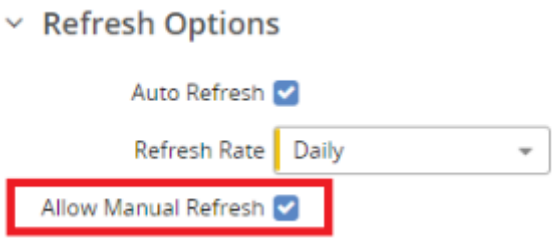
- Manual Refresh
- Cache Warming
- Scheduled Refresh

Manual Refresh

Depending on the configuration of each KPI, it is possible for Action Center users to manually force a refresh of the data from the OpenEdge database by selecting the refresh icon in the lower right corner of a dashboard panel. The icon is displayed next to a date-time stamp showing when the data was last retrieved from the operational database, as highlighted in the following graphic.



The refresh icon is only available if the Allow Manual Refresh flag in the KPI screen is selected for the KPI associated with the panel.



When the refresh icon is selected, the source browse or FRW KPI is re-processed, re-cached in memory, and displayed in the Action Center. All the Action Center panels are re-displayed in the browser, but only those panels using the refreshed browse display refreshed source data.

> ⚠ In the case of KPIs whose data source is a large browse, a manual refresh can take several minutes, because the source browse must be processed, the results stored in Cassandra and re-cached in Spark, and the visuals in the Action Center re-rendered by Logi Info.

Because of the potential load in the system when KPIs use large browses, manual refreshes should only be used when current data is required. Routine regular refreshes should be accomplished by enabling *cache warming* and *scheduled refresh*.

The following system-level properties affect KPI caching and refresh behavior, enforcing limits to ensure acceptable online performance.

Property	Purpose/Description	Default Value	Tuning Considerations
qad-analytics-core.maxKpiActiveFields	Maximum number of active fields that may be included in a KPI.	20	Should be limited in order to conserve system resources.
qad-analytics-core.browseBatchRefreshTimeLimit	Causes the copying of a KPI result set into the data lake to be skipped if the same KPI was copied more recently than the value of this property, expressed in minutes.	1	Normally should not be changed.

Cache Warming

Cache warming refers to the process of caching all the KPIs in memory when an environment is started. This allows the KPIs to be available without a significant wait for an Action Center to display the first time one of the underlying KPIs is needed. By default, cache warming is enabled when the system is installed.

In the case of browse-based KPIs cached in the Query Service, it is important to note that cache warming does not necessarily refresh the cache contents to reflect current OpenEdge database contents. If the required browses are already present in the Cassandra data lake, cache warming loads the memory caches from Cassandra without reprocessing the browse requests. Only required browses that are not yet in Cassandra are retrieved from the OpenEdge databases through browse requests. This approach allows cache warming to proceed much faster and with lower system resource usage at application startup, which is often an important practical consideration. Refreshing the caches from the operational database sources is accomplished mainly by scheduled refresh.

Cache warming is controlled by various properties that can be modified using YAB if necessary. This document does not provide a comprehensive list, but only covers those that are most likely to affect overall performance and/or require tuning.

Browse KPI Cache Warming Properties

Property	Purpose /Description	Default Value	Tuning Considerations
qad-analytics-core.cache.kpis-browse.loadAtStartup	Indicates if the cache is warmed at application startup.	true	Set to false in order to disable cache warming.
qad-analytics-core.metricKpiCacheWarmerExecutor.poolSize	Number of KPI refreshes that can be requested concurrently.	2	Increase to submit refresh requests faster to the Query Service, potentially warming the cache in less time but at the cost of greater resource usage. Must have a value of 1 or greater. The effect of this setting is constrained by the Query Service property qad-qracore.browseCassandraDataService.concurrency (described below).

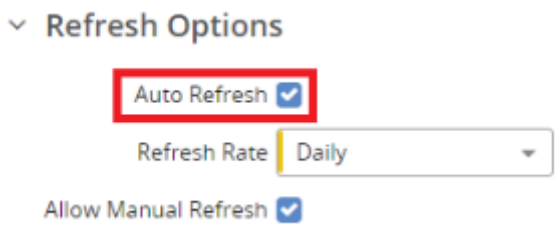
FRW KPI Cache Warming Properties

Property	Purpose /Description	Default Value	Tuning Considerations
qad-analytics-financials.cache.kpis-frw.loadAtStartup	Indicates if the cache is warmed at application startup.	true	Set to false in order to disable cache warming.
qad-analytics-financials.frwKpiCacheWarmerExecutor.poolSize	Number of KPI refreshes that can be requested concurrently.	2	Increase to submit refresh requests faster to the Query Service, potentially warming the cache in less time but at the cost of greater resource usage. Must have a value of 1 or greater.

Scheduled Refresh

Because Action Center displays retrieve their KPI data from in-memory caches, the data may not reflect current database contents. If the data sets are not periodically refreshed, over time they will become stale and less relevant to the needs of the organization. While end users can trigger the data in particular Action Center panels to be refreshed from the OpenEdge sources, manual refreshes are resource intensive and often slow. To keep the Action Center contents current enough to be useful, the system automatically refreshes the browse and FRW KPI caches periodically, based on a configurable schedule. This feature is called 'scheduled refresh.'

Scheduled refresh is configurable by KPI. KPIs can be explicitly enabled for scheduled refresh on a daily, weekly, or monthly basis in the KPI screen using the 'Auto Refresh' setting.



However, there is a limit on the number of KPIs for which scheduled refresh can be enabled, as described below.

Property	Purpose /Description	Default Value	Tuning Considerations
qad-analytics-core.maxKpisAutoRefreshed	Maximum number of KPIs for which scheduled refresh may be enabled	30	Can be increased to allow more KPIs to be automatically refreshed, at the cost of more resource-intensive browse requests. The impact of a longer scheduled refresh depends on the number of KPIs, the size of browse result sets, and the overall system load at the time of day when the scheduled refresh is run.

Scheduled refresh is controlled by various properties that can be modified using YAB if necessary. This document does not provide a comprehensive list, but only covers those that are most likely to affect overall performance and/or require tuning.

The scheduled refresh process is started and runs inside the tomcat-webui instance, not in separate scripts. If tomcat-webui is not running at the time when the scheduled refresh is scheduled (ex. during an offline backup), or was stopped before an in-process scheduled refresh could complete, no special recovery process is initiated. Instead, the scheduled refresh will run at the next scheduled date-time once tomcat-webui is running again.

Browse KPI Scheduled Refresh Properties

Property	Purpose/Description	Default Value	Tuning Considerations
qad-analytics-core.cache.kpis-browse.scheduledRefresh.enabled	Enables scheduled refresh processing across the system, based on the other properties and individual KPI configuration.	true	Set to false to disable all scheduled refreshes.
qad-analytics-core.cache.kpis-browse.scheduledRefresh.batchSize	Maximum number of requests that are batched for processing by a single KPI refresh thread.	100	Lower values may allow the scheduled refresh to be completed faster, at the cost of using more memory and/or processor resources. Not recommended to change.
qad-analytics-core.cache.kpis-browse.scheduledRefresh.quartzJobCount	Number of background threads that can concurrently request KPI refreshes. Must be greater than or equal to 2.	2	More threads would allow more scheduled refreshes to run concurrently, at the cost of using more memory and/or processor resources.
qad-analytics-core.cache.kpis-browse.scheduledRefresh.cronExpression	String expression in a cron format that specifies schedule when the scheduled refresh is run. See the Quartz documentation for a more detailed description of cron syntax. This setting does not cause cron scripts to run. The cron syntax is only used to set the schedule for background refresh activity run within the Tomcat environment.	0 0 0 * * ? (every day at midnight)	Set to a time schedule that suits the organization, based on system load and user activity over a 24-hour period. If possible, schedule for a time of day with low online user activity.
qad-analytics-core.cache.kpis-browse.scheduledRefresh.kpi-weekly-day	For KPIs that are configured to be refreshed weekly, specifies the day of the week on which the refresh is performed. Valid values are sun, mon, tue, wed, thu, fri, sat.	sun	Set to a day of the week that suits the organization, based on system load and user activity.
qad-analytics-core.cache.kpis-browse.scheduledRefresh.kpi-monthly-day	For KPIs that are configured to be refreshed monthly, specifies the day of the month on which the refresh is performed. Syntax is a comma-separated string with two values: • first or last: Indicates if the refresh is defined relative to the beginning or end of each month. • offset days: Sets the number of days before or after the first or last of the month to perform the refresh. A positive value states the number of days after the first or last day of the month; a negative value states the number of days before the first or last day of the month; and a value of 0 indicates the first or last day of the month with no offset.	first,0	Set to a day of the month that suits the organization, based on system load and user activity.
qad-analytics-core.browseBatc	Minimum number of minutes allowed between refreshes of the same KPI browse by the Query Service. If a requested browse was refreshed within this time interval, the new request is	1	Increase the value in order to increase the minimum amount of time allowed for refreshes of the same browse. This may be important in order to prevent excessive load on the system, in terms of AppServer

hRefreshTimeLimit	skipped. This property prevents repeated refreshes of the same browse from being processed within a short time interval (for example, by repeatedly clicking the refresh control inside an Action Center panel).		agents and SQL connections that are consumed processing browse requests. Smaller values allow the same browse to be refreshed more frequently. This property allows you to adjust the trade-off between system resource usage and data currency of Action Center displays.

FRW KPI Scheduled Refresh Properties

Property	Purpose/Description	Default Value	Tuning Considerations
qad-analytics-financials.cache.kpis-frw.scheduledRefresh.enabled	Enables scheduled refresh processing across the system, based on the other properties and individual KPI configuration.	true	Set to false to disable all scheduled refreshes.
qad-analytics-financials.cache.kpis-frw.scheduledRefresh.batchSize	Maximum number of refresh requests that are batched for processing in a single thread.	100	Lower values may allow the scheduled refresh to be completed faster, at the cost of using more memory and/or processor resources. Not recommended to change.
qad-analytics-financials.cache.kpis-frw.scheduledRefresh.quartzJobCount	Number of background threads that can concurrently request KPI refreshes. Must be greater than or equal to 2.	2	More threads would allow more scheduled refreshes to run concurrently, at the cost of using more memory and/or processor resources.
qad-analytics-financials.cache.kpis-frw.scheduledRefresh.cronExpression	String expression in a cron format that specifies schedule when the scheduled refresh is run. See the Quartz documentation for a more detailed description of cron syntax.	0 0 0 * * ? (every day at midnight)	Set to a time schedule that suits the organization, based on system load and user activity over a 24-hour period. If possible, schedule for a time of day with low online user activity.
qad-analytics-financials.cache.kpis-frw.scheduledRefresh.kpi-weekly-day	For KPIs that are configured to be refreshed weekly, specifies the day of the week on which the refresh is performed. Valid values are sun, mon, tue, wed, thu, fri, sat.	sun	Set to a day of the week that suits the organization, based on system load and user activity.
qad-analytics-financials.cache.kpis-frw.scheduledRefresh.kpi-monthly-day	For KPIs that are configured to be refreshed monthly, specifies the day of the month on which the refresh is performed. Syntax is a comma-separated string with two values: • first or last: Indicates if the refresh is defined relative to the beginning or end of each month. • offset days: Sets the number of days before or after the first or last of the month to perform the refresh. A positive value states the number of days after the first or last day of the month; a negative value states the number of days before the first or last day of the month; and a value of 0 indicates the first or last day of the month with no offset.	first,0	Set to a day of the month that suits the organization, based on system load and user activity.

Key Query Service Properties

The following properties are specific to the Query Service, and therefore affect the caching of only browse-based KPIs, not FRW KPIs. They can affect both cache warming and scheduled refresh processing. This is not a comprehensive list, but only covers those properties that are most likely to affect overall performance and/or require tuning.

Property	Purpose /Description	Default Value	Tuning Considerations
qad-qracore.	Number of browse	3	Limits the number of concurrent browse requests that can be processed by the Query Service. Browse requests are one of the following types.

browseCassandraDataService.concurrency	retrievals and data copies into Cassandra that can be processed concurrently.		• SQL queries using an OpenEdge JDBC connection • Browse engine queries run on a QRA AppServer agent. This property limits the number of browse requests of either type that can be processed at the same time. For browse requests processed on the QRA AppServer, this is a critical setting, as Progress AppServers are often a scarce and CPU-intensive system resource. If the number of AppServer agents being used to process Query Service browse requests causes the environment to run out of available agents, a fatal 'No Servers Available' error may be raised by Progress. This error will cause the request that needs the AppServer agent to fail, whether it comes from the Query Service or from another system component. For this reason, this property should be set to a value lower than the total number of QRA AppServer agents available in the environment, allowing enough agents available for other activities to proceed while cache warming or a scheduled refresh is in process.
qad-qracore.browseCassandraDataService.timeoutLimit	Maximum number of seconds that the Query Service will wait for a page of output to be returned from a browse request.	120	In a heavily loaded environment with high CPU and/or data retrieval activity, especially using AppServer agents, this setting might have to be increased to give the system time to retrieve complete browse results.
qad-qracore.browseCassandraDataService.pageSize	Number of records retrieved in a single page or chunk from a browse request.	5000	Could be adjusted to retrieve browse data from the source in smaller or larger chunks, potentially affecting data retrieval traffic and latency.

Monitoring Cache Refreshes

There is no console or screen to monitor the progress of cache warming or scheduled refreshes. However, a portion of the Query Service activity can be viewed as a list of jobs shown in the Spark web UI, briefly introduced in Action Center Maintenance and Troubleshooting. In addition, progress messages are written to the tomcat-webui console log when the appropriate settings are configured in the logback.xml file. Instructions for configuring logging, either manually or through YAB, are not covered in this document. However, the settings necessary in logback.xml to obtain comprehensive KPI caching messages are summarized below.

```
    <logger name="com.qad.analytics.core.service.impl.KpiCacheLoaderBase" level="debug"
additivity="false">
        <appender-ref ref="stdout" />
    </logger>
    <logger name="com.qad.analytics.core.service.impl.BrowseKpiCacheLoader" level="debug"
additivity="false">
        <appender-ref ref="stdout" />
    </logger>
    <logger name="com.qad.analytics.financials.service.impl.FrwKpiCacheLoader2" level="debug"
additivity="false">
        <appender-ref ref="stdout" />
    </logger>
```

For these settings, it is assumed that the appender *stdout* references the standard tomcat-webui console log file. The resulting messages have a log level of DEBUG. They trace the caching of each KPI for each domain or financial entity, without showing details about the related browses or queries.

'View-As User' Feature

Starting with the March 2020 release, it is possible for a Web UI user to display an Action Center in the same way that it would be displayed to a different Web UI user who is subordinate in the organization to the actual user. This feature allows a manager, for example, to easily see the same Action Center data that one of his or her subordinates would see. Special configuration in the Web UI is required to enable this feature in a secure way. The set up and use of the feature is described in the online help and Adaptive UX User Guide.

However, using 'View-As User' can decrease the online performance of the Action Centers and requires additional security configuration. In addition, in many customer environments the feature will not be used at all. Therefore, it has been disabled by default at the system level using the following Web UI property.

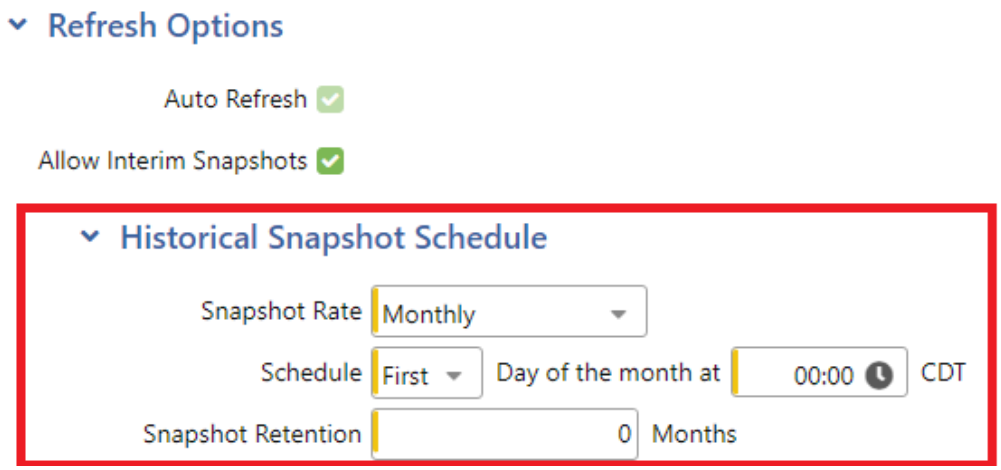
```
qad-analytics-core.logips.viewAs.enabled=false
```

In order to configure and use 'View-As User' permissions in the Web UI, the feature must first be enabled. To do this, set the property to true using YAB. It will be activated after the next YAB update and tomcat-webui restart.

'Historical KPIs' Feature

Starting with the September 2020 release, historical KPIs were introduced. Historical KPIs allow you to track data trends by comparing summary-level metrics over time. In the KPIs view, you can create historical KPIs and configure the system to take snapshots of your KPI for particular periods. You can then compare these periods to analyze the data.

Like the Scheduled and Manual Refresh functions, historical snapshots are configured by KPI on the KPI screen.



However, there are limits on the number and size of snapshots that are configured at the system level, as described below.

> Because historical KPI snapshots accumulate over time based on their frequencies and schedules, they can grow to consume large amounts of disk space depending on their size, grouping level, and so on. The limits on historical KPI snapshots should therefore be considered carefully for each Adaptive UX installation.

Historical KPI Properties

Property	Purpose/Description	Default Value	Tuning Considerations
qad-analytics-core.kpi.historical.enabled	Determines if historical KPI functionality is enabled.	true	
qad-analytics-core.kpi.historical.schedule.enabled	Indicates if the historical KPI schedule is enabled. This property depends on 'qad-analytics-core.kpi.historical.enabled' and if it is disabled, then the historical KPI schedule is also disabled despite the value of the property.	true	
qad-analytics-core.kpi.historical.maxSnapshots	Maximum number of snapshots in the system.	1100	
qad-analytics-core.kpi.historical.	Maximum number of records that may be retrieved for a single Historical KPI snapshot.	200000	

maxRowCount			
qad-analytics-core.kpi.historical.maxKpiActiveFields	Maximum number of fields or data columns that may be enabled for a single Historical KPI.	20	
qad-analytics-core.kpi.historical.cassandra.gc_grace_seconds	Delay of final deletion of tombstones in Historical Data tables in Cassandra.	0	A value of zero causes tombstones to be deleted immediately, which is good in a single-node cluster. However, in a multi-node cluster a value of zero would cause data integrity problems across the nodes in the cluster.

Special Logi Info Configuration

In releases prior to September 2019 when all Action Centers were supported using Logi Info, there were several situations that required manual configuration changes to the Logi Info settings. The following information is not necessary for environments created with the September 2019 release or later.

Special Calendar Support

Fiscal Year and Quarter

For organizations that use a fiscal calendar different from the standard calendar, Action Centers in Logi Info allow time-line charts to be created that present dates in terms of fiscal year or fiscal quarter, as well as calendar year or quarter. However, the fiscal periods used by Logi Info are not integrated with the fiscal calendar used in Adaptive Applications. In order to see the Fiscal Year and Fiscal Quarter options when configuring visuals for Action Centers, you must manually add a property to the main Logi Info configuration file.

The name of this file is _Settings.lgx. It is saved in the _Definitions/ directory under the root directory of the qad-dashboards webapp. To find this root directory, run the following command.

```
yab config webapp.analytics-logi.dir
```

Open the file with a text editor. Find the Globalization element near the end of the file.

```
<?xml version="1.0" encoding="utf-8"?>
<Setting ID="_settings">
    <GlobalCSS StyleSheet="QAD.Qracore.actioncenter.css" Theme="ChannelIslands"/>
    <Application/>
...
    <Globalization UserCulture="@Session.UserDisplayLocale~"/>
    <ideTestParams/>
</Setting>
```

Add the attribute 'FirstDayOfFiscalYear' to the Globalization element as shown below. Set its value to MM/DD, where MM is the two-digit month of the calendar year and DD is the two-digit day of the month.

```
<?xml version="1.0" encoding="utf-8"?>
<Setting ID="_settings">
    <GlobalCSS StyleSheet="QAD.Qracore.actioncenter.css" Theme="ChannelIslands"/>
    <Application/>
...
    <Globalization FirstDayOfFiscalYear="02/01" UserCulture="@Session.UserDisplayLocale~"/>
    <ideTestParams/>
</Setting>
```

Save the change. It will take effect immediately, with no need to restart the environment.

If this property is not present in the file, Logi Info assumes that the fiscal and calendar years are the same, with no need for the Fiscal Year or Fiscal Quarter options when configuring visuals.

Week Start Day

Action Centers in Logi Info allow time-line charts to be created that present dates in one-week blocks, by selecting the Week option when configuring the chart. However, some organizations prefer to express weeks with non-standard start and end days, rather than Sunday through Saturday. In order to set the start day of the week for purposes of creating visuals, you must manually add a property to the main Logi Info configuration file.

The name of this file is _Settings.lgx. It is saved in the _Definitions/ directory under the root directory of the qad-dashboards webapp. To find this root directory, run the following command.

```
yab config webapp.analytics-logi.dir
```

Open the file with a text editor. Find the Globalization element near the end of the file.

```
<?xml version="1.0" encoding="utf-8"?>
<Setting ID="_settings">
    <GlobalCSS StyleSheet="QAD.Qracore.actioncenter.css" Theme="ChannelIslands"/>
    <Application/>
...
    <Globalization UserCulture="@Session.UserDisplayLocale~"/>
    <ideTestParams/>
</Setting>
```

Add the attribute 'FirstDayOfWeek' to the Globalization element as shown below. Set its value to one of the following.

- 0 = Sunday (the default value)
- 1 = Monday
- 2 = Tuesday
- 3 = Wednesday
- 4 = Thursday
- 5 = Friday
- 6 = Saturday

```
<?xml version="1.0" encoding="utf-8"?>
<Setting ID="_settings">
    <GlobalCSS StyleSheet="QAD.Qracore.actioncenter.css" Theme="ChannelIslands"/>
    <Application/>
...
    <Globalization FirstDayOfWeek="1" UserCulture="@Session.UserDisplayLocale~"/>
    <ideTestParams/>
</Setting>
```

Save the change. It will take effect immediately, with no need to restart the environment.

Action Center Maintenance and Troubleshooting

Backup and Restore Activities

This section summarizes considerations for regular data backup and restore activities that are specific to Action Centers and the Query Service.

Logi Platform Services Product Database

Starting with the September 2019 release, the Action Center dashboard and visual definitions developed using Logi Platform Services are stored in Logi's Product Database (PDB), which is implemented using the H2 relational database manager. These files are changed online as a result of end-user activity that creates, modifies, or deletes KPIs, Action Centers, and visuals. The PDB also contains much configuration data internal to LogiPS. PDB activity is not recorded in OpenEdge database rollback or roll-forward logs. As a result, the PDB cannot be precisely synchronized with the OpenEdge databases when the databases must be restored to a particular point in time through automatic roll-forward operations, such as in some disaster recovery scenarios. In practice, the risk of data corruption is relatively low, given the following points.

- KPI, Action Center, and visuals maintenance are typically low-volume, low-frequency activities that are performed by only a subset of users during working hours.
- The contents of the PDB are not tightly coupled with OpenEdge database contents. As a result, changes to the Logi files generally do not affect the database and vice versa.

To minimize the risk of data loss, the PDB should always be backed up at the same time as the OpenEdge databases. The following YAB command creates a backup of the Logi Platform Services PDB.

```
yab logi-platform-services-default-backup
```

The backups are stored in the directory referenced by the following YAB property.

```
logi-platform-services-backup.dir
```

The following YAB commands include the PDB in backups of the system environment.

```
yab environment-online-backup
yab environment-offline-backup
```

The following YAB command provides a list of all the existing PDB backups for Logi Platform Services.

```
logi-platform-services-default-backup-list
```

The output of the backup list command is similar to the following example.

```
Tag: default
--------------
Location: /dr01/dbs/backup/default/logi-platform-services

default.zip                          May 27, 2019 8:05:27 AM

Tag: 20190527080605
--------------
Location: /dr01/dbs/backup/20190527080605/logi-platform-services

default.zip                          May 27, 2019 8:06:06 AM

Tag: foo
--------------
Location: /dr01/dbs/backup/foo/logi-platform-services

default.zip                           May 27, 2019 8:04:40 AM
```

The following YAB command deletes all Logi Platform Services PDB backups.

```
yab logi-platform-services-backup-remove
```

It is executed as part of the following YAB command.

```
yab database-all-backup-remove
```

The PDB can be restored only when Logi Platform Services is offline. It is restored by the following YAB command.

```
yab logi-platform-services-default-restore
```

The restore is also executed as part of the following YAB command.

```
yab environment-restore
```

Because the PDB contains license information, if a PDB backup from an environment containing a different Logi Platform Services license is restored into the target environment, the correct license should be re-imported after the restore using the following YAB command.

```
yab logi-platform-services-default-license-import
```

Logi Files

Prior to the September 2019 release, the Action Center dashboard and visual definitions were stored in XML files inside the Logi Info web application, not in a database. For environments still running a pre-September 2019 release, these files are changed online as a result of end-user activity that creates, modifies, or deletes KPIs, Action Centers, and visuals. Changes to these files are not recorded in OpenEdge database rollback or roll-forward logs. As a result, these files cannot be precisely synchronized with the OpenEdge databases when the databases must be restored to a particular point in time through automatic roll-forward operations, such as in some disaster recovery scenarios. In practice, the risk of data corruption is relatively low, given the following points:

- KPI, Action Center, and visuals maintenance are typically low-volume, low-frequency activities that are performed by only a subset of users during working hours.
- The contents of the Logi files are not tightly coupled with OpenEdge database contents. As a result, changes to the Logi files generally do not affect the database and vice versa.

All the Action Center files are stored in a single directory inside the Logi Info web app. To find the directory location, query the YAB configuration.

```
yab config qad-analytics-core.logiDashboardsPath
```

To minimize the risk of data loss, the Logi files should always be backed up at the same time as the OpenEdge databases. The following YAB commands include the Logi files in backups of the system environment.

```
yab environment-online-backup
yab environment-offline-backup
yab directorybackup-backup
yab directory-action-center-backup
```

Data restore activities performed by system administration personnel should be implemented to include both the OpenEdge databases and the Logi files. The files can be backed up and restored through simple file copies, without the use of any special utilities. The following YAB commands restore the Logi files backups created by a previous YAB backup.

```
yab directorybackup-restore
yab directory-action-center-restore
```

Because changes to the Logi files are not automatically logged, QAD recommends that the Logi files be backed up more frequently than the OpenEdge databases in order to support recovery procedures when it is necessary to restore the entire system to a stable state as of a particular point in time. If the OpenEdge databases must be restored and rolled forward to a particular point in time, more frequent Logi file backups allow the files to be restored to a state that is closer to the restored database state. There is still the possibility of some Action Center data loss if the current Logi files were lost during a severe service outage, but the risk can usually be managed to a low level through this approach.

Cassandra Keyspaces

The 'browses' keyspace in the Cassandra data lake that contains Query Service data is not a system of record, but is created entirely from the contents of operational OpenEdge database tables. There is no need to back up its contents or restore a backup in case of data loss. It should therefore be included in the list of keyspaces exempted from backups in the YAB property cassandra.default.node.backup.blacklist. Whenever there is a need to restore/refresh the data in this keyspace to the current state, the keyspace should be rebuilt from its sources as described in the section 'Rebuilding the Cassandra Keyspace.'

The 'historical_kpi' keyspace in the Cassandra data lake is the system of record for historical KPI snapshots, and must be included in all database backups. It should therefore be omitted from the list of keyspaces exempted from backups in the YAB property cassandra.default.node.backup.blacklist.

Spark Cache

Within the Query Service, Spark is used as an on-demand, memory-based cache of browse data that is loaded from the Cassandra data lake. Hence, there is no need to back up its contents or restore a backup in case of data loss. Instead, the cache is rebuilt or refreshed at the same time as the Cassandra keyspace (see above).

Log Files

Because much of the Action Center and Query Service processing is performed in the background and is not visible in the user interface, log files are the most important resource for diagnosing problems.

Tomcat Logs

The console log file written by the Tomcat instance that supports the Web UI (tomcat-webui) is the first place to look for error details. By default, the current log file is named catalina.out, and the files created on previous days are named catalina.<date>.log, where <date> is the date when the log was written. These file are stored in the logs/directory under the Tomcat instance. To find the root directory of the Tomcat instance, run the following command.

```
yab config tomcat.webui.base
```

Logi Platform Services Logs

To find the Logi Platform Services log files generated by the Logi Data Service, query the YAB configuration.

```
yab config logi-platform-services.default.log.dir
```

The log files in this location are as follows:

- lps-default-dataservice.log: describes starting and stopping DataServices and H2 services, and displays errors with the services.
- lps-default-dataservice-error.log: describes things like JWS errors and other service errors.

To find the Logi Platform Services log files generated by the Logi Application Service, query the following property and go to the logs/ sub-directory under it.

```
yab config logi-platform-services.default.dir
```

The log files in this location are as follows:

- logiApplicationService.<date>.log: describes possible connection errors.
- logiDataServiceStartup.log: describes backup locations and related java information.
- microservice-<date>.log: describes errors in Logi's microservice processes (example: PDF export actions).
- licenseImport log: describes the license import processes.
- clientSecret.log: describes the process that assigned client secret codes.

Cassandra Logs

To find the Cassandra log files, query the YAB configuration.

```
yab config cassandra.default.node.jvm.logs.dir
```

The log files are located in the sub-directory default/ within this directory. The cassandra-default.log file shows Cassandra activity, and the gc.log.* files show its Java garbage collection activity.

Spark Logs

To find the Spark log files, query the YAB configuration for the master and slave processes running in Spark.

```
yab config spark.masterdefault.env.spark.log.dir
```

The Spark master process manages the resource used by Spark workers to process particular requests.

```
yab config spark.slavedefault.env.spark.log.dir
```

Spark worker processes carry out particular tasks based on the incoming requests.

YAB Logs

If errors related to Action Centers or the Query Service are raised while running a YAB command, consult the YAB log file for details recorded by YAB (for example, a cache warming failure when the Tomcat instance is started during a YAB update).

The YAB log file is named yab.log, and is stored in the build/logs/ directory under the root of the installation.

Cassandra Shell

Cassandra provides a command-line shell *cqlsh* that can be used to execute CQL commands, the SQL-like language supported by Cassandra. The shell is especially useful for examining the contents of the browses or historical_kpi keyspaces, which contain all the Query Service browse data retrieved from the OpenEdge databases.

To run the Cassandra shell, Python 2.7 must be present on the command PATH.

To find the location of the shell utility and the Cassandra port number to which to connect, query the YAB configuration.

```
yab config cassandra.default.install.scripts.dir
```

```
yab config cassandra.default.node.main.native_transport_port
```

Change to the directory containing the Cassandra scripts. Start the shell, passing the required hostname and port number. The actual hostname (not 'localhost') must be used.

```
cd <Cassandra scripts directory>
./cqlsh <hostname> <Cassandra port>
```

Determine the keyspace whose data you want to review, and make this keyspace the default for all subsequent CQL commands.

```
cqlsh> use browses;
```

Once the shell is running, it can be used to query the contents of the data lake with various commands, such as the following.

List the browse tables in the browses keyspace

The list shows the browse tables stored in the data lake. The list includes the domains or financials entities associated with the browse tables.

```
cqlsh:browses> describe tables;

kpi_357844472_12mex      kpi_1688405265_20fra     kpi__1355073026_10usa
kpi__846170097_12mex     kpi_1852174264_20fra     kpi__1863008167_22uk
kpi_1988561950_21nl      kpi_1988561950_12mex     kpi__692383936
kpi__1809347128_20fra    kpi__45865388_23ger      kpi__1451599441_10usa
kpi_1202448307_12mex     kpi_1044215493_21nl      kpi_1533800352_30chn
kpi_1202448307_30chn     kpi__1063222905_12mex    kpi_1384866991_10usa
kpi__186238821_12mex     kpi_1705405443_qad       kpi_1988561950_20fra
kpi_1650412229_11can     kpi__1798778746_20fra    kpi__1063222905_11can
kpi__1975819738_20fraco  kpi_1471838795_31aus     kpi_1105371684_10usa
kpi_1206509372_31aus     kpi_1384866991_11can     kpi__1423034166_21nl
kpi__1282164377_23ger    kpi__45865388_31aus      kpi__1809347128_12mex
kpi__1422684978_23ger    kpi__1319591205_22uk     kpi_642442275_10usaco
kpi__1942540096_12mex    kpi__1808255083_10usa    kpi__1604144893_31aus
kpi_1533800352_10usa     kpi__1423034166_11can    kpi__875669433
kpi__78780668_12mexco    kpi_1044215493_31aus     kpi_1529480452_31aus
kpi_1101437247_31aus     kpi_1175672099_12mex     kpi_1852174264_11can
kpi_1854608121_30chn     kpi_1471838795_21nl      kpi_1206509372_11can
kpi__1916630248_11can    kpi_1101437247_40brz     kpi_131336283_21nl
kpi_1080786763           kpi_131336283_22uk       kpi_220484148_12mex
kpi_514817284_10usa      kpi_1529480452_22uk      kpi__6799222_40brz
...

cqlsh:browses>
```

As of the September 2020 release of Adaptive UX, the KPI and browse cannot be identified based on the table name. The table names may be changed in future releases to facilitate easier inspection using the Cassandra shell.

Display status information about each table in the browses keyspace

This information comes from the Query Service table browsecopystatus. It contains the date-time when the results of each KPI browse were last copied into Cassandra.

```
cqlsh:browses> select * from browsecopystatus;

browsecopy                                    | amountcopied | canceled |
failed                          | finished                          |
started                         | submitted
-----------------------------------------------+--------------+----------
+---------------------------------+---------------------------------
+---------------------------------+---------------------------------
   e874314d-425a-59b6-6314-c157204f2414_31aus_ |            0 |     null
|                                 null | 2020-08-23 00:49:06.683000+0000 | 2020-08-23 00:49:04.681000
+0000 | 2020-08-23 00:48:55.330000+0000
    988f1fb8-87a7-0cad-5714-ef2780ae4474_22uk_ |           64 |     null
|                                 null | 2020-08-22 23:09:37.249000+0000 | 2020-08-22 23:09:35.410000
+0000 | 2020-08-22 23:09:27.684000+0000
   d1be0cc5-4c16-6c9b-6314-4e6830e1e6ed_20fra_ |            0 |     null
|                                 null | 2020-08-23 00:46:22.606000+0000 | 2020-08-23 00:46:20.917000
+0000 | 2020-08-23 00:46:13.553000+0000
    ec8e0e0b-1866-2d80-5514-a431c803afa0_21nl_ |          747 |     null
|                                 null | 2020-08-22 23:23:44.525000+0000 | 2020-08-22 23:23:40.774000
+0000 | 2020-08-22 23:23:40.127000+0000
   969a714f-4089-b4a1-5414-c84af01bc680_40brz_ |          819 |     null
|                                 null | 2020-08-22 23:08:23.896000+0000 | 2020-08-22 23:08:16.922000
+0000 | 2020-08-22 23:08:09.055000+0000
   862404b7-980e-8885-5414-5663a00cee43_30chn_ |         4251 |     null
|                                 null | 2020-08-22 23:03:03.404000+0000 | 2020-08-22 23:02:45.261000
+0000 | 2020-08-22 23:02:09.957000+0000
   cf4f03ad-e23f-5a99-6214-56f5481b6a0c_30chn_ |            0 |     null
|                                 null | 2020-08-23 00:46:41.284000+0000 | 2020-08-23 00:46:39.181000
+0000 | 2020-08-23 00:46:32.473000+0000
   e3a23bcc-e989-7890-5814-7b9a40ce2e87_10usa_ |            1 |     null
|                                 null | 2020-08-22 23:21:45.147000+0000 | 2020-08-22 23:21:44.142000
+0000 | 2020-08-22 23:21:43.601000+0000
   a84c4040-874b-9c98-5514-267450caeec0_23ger_ |          189 |     null
|                                 null | 2020-08-22 23:12:18.824000+0000 | 2020-08-22 23:12:17.125000
+0000 | 2020-08-22 23:12:12.562000+0000
   ce6200b2-b081-75a5-6b14-c987c0b5eeb6_10usa_ |            0 |     null
```

```
|                          null | 2020-08-22 23:18:53.094000+0000 | 2020-08-22 23:18:51.696000
+0000 | 2020-08-22 23:18:50.636000+0000
   dbf98bf0-4d3e-b48d-5614-df4c980b1c9b_10usa_ |           65 |     null
|                          null | 2020-08-22 23:19:09.521000+0000 | 2020-08-22 23:19:08.502000
+0000 | 2020-08-22 23:19:07.997000+0000
   862404b7-980e-8885-5414-cfa1708c494f_40brz_ |         3435 |     null
|                          null | 2020-08-22 23:03:30.869000+0000 | 2020-08-22 23:03:20.566000
+0000 | 2020-08-22 23:02:10.463000+0000
   ec8e0e0b-1866-2d80-5514-b522e05c709e_12mex_ |            0 |     null
|                          null | 2020-08-22 23:24:28.509000+0000 | 2020-08-22 23:24:20.272000
+0000 | 2020-08-22 23:23:46.528000+0000
   ebce0a62-120c-bfa7-5714-023820b2c450_11can_ |         4120 |     null
|                          null | 2020-08-22 23:23:00.680000+0000 | 2020-08-22 23:22:45.467000
+0000 | 2020-08-22 23:22:27.670000+0000
   ba98dc66-9607-7698-5514-4e22e044cf41_30chn_ |         4452 |     null
|                          null | 2020-08-22 23:17:19.834000+0000 | 2020-08-22 23:17:06.574000
+0000 | 2020-08-22 23:15:40.269000+0000
 b7e49d22-1d54-81b5-5814-ea8a181d87df__10usaco |            0 |     null | 2020-08-26 17:33:
41.945000+0000 |                           null | 2020-08-26 17:33:28.697000+0000 | 2020-08-26
17:32:59.894000+0000
   e29f724f-acba-ed81-5914-c51ef00b477b_40brz_ |         2425 |     null
|                          null | 2020-08-22 23:21:11.908000+0000 | 2020-08-22 23:21:04.490000
+0000 | 2020-08-22 23:20:39.019000+0000
...

cqlsh:browses>
```

Remove the browse data

If the browse data in the Query Service ever became corrupted and needed to be entirely refreshed, no backup-restore procedure would be needed. Instead, the Cassandra data lake can be emptied and re-populated by processing the necessary browses against the current OpenEdge source databases.

Individual browse tables can be dropped from browses keyspace as follows.

```
cqlsh:browses> drop table <table name>;
```

Here is an example.

```
cqlsh:browses> drop table kpi__186238821_10usa;
```

Alternatively, the following command will drop all browse tables from the data lake. A timeout error may be displayed if processing requires more that the default 10-second limit. However, in this case the command will still be processed in the background despite the timeout.

```
cqlsh:browses> drop keyspace browses;
```

To re-populate the Cassandra keyspace with browse data, restart the Tomcat Web UI instance with cache warming enabled. All browses needed to support the KPIs used by the Action Centers are re-processed, and the results are copied into Cassandra.

To exit the shell, use the quit command.

```
cqlsh:browses> quit;
```

For more information about the shell and other Cassandra tools, see the documentation at the Apache Cassandra web site.

Spark Web User Interface

Spark provides a web user interface that displays historical information about the work it has performed for the Query Service. The display includes:

- A list of scheduler tasks and stages
- A summary of data set sizes and memory usage
- Environmental information

- Information about the running Spark executors

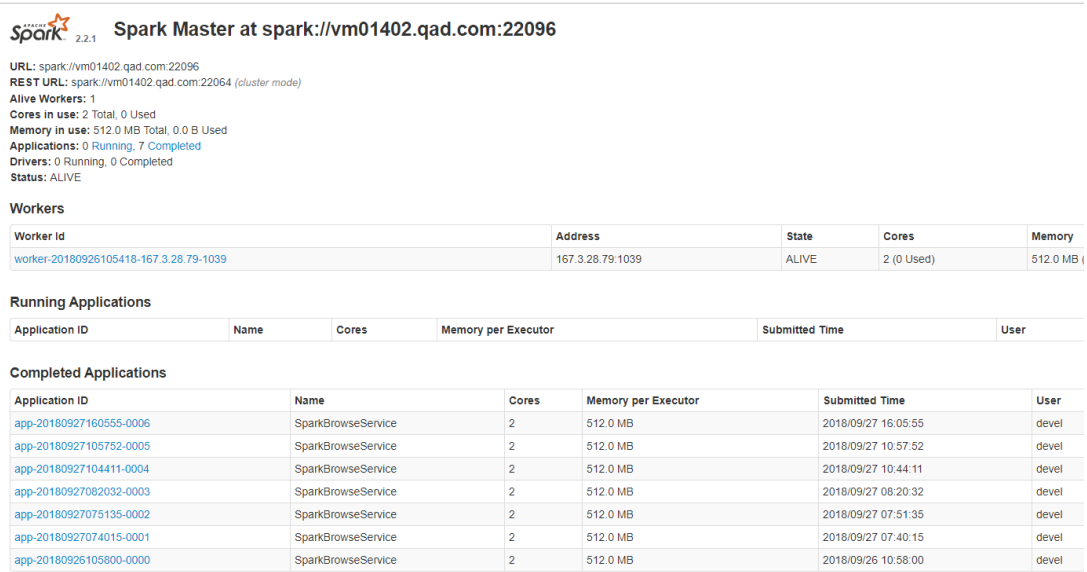
For more information, see the documentation at the Apache Spark web site.

To display the UI, query the YAB configuration to find the correct HTTP port to connect.

```
yab config spark.masterdefault.env.webui.port
```

In a web browser, go to the page http://<hostname>:<Spark Web UI port>

Below is a sample Spark display.



The page contains hyperlinks providing more details on particular tasks and links to native Spark log entries for them.

Spark Beeline Tool

While the Spark Web UI described above provides many operational details regarding Spark activities, queries, and environment characteristics, it does not provide a way to query the browse data that has been cached in the Query Service. However, Spark also includes beeline, a command-line utility that can connect to the Spark ThriftServer within the Query Service, and access the cached data using SQL.

To find the location of the Spark files, the ThriftServer port number to which to connect, and the required ThriftServer credentials, query the YAB configuration.

```
yab config packages.spark.dir
```

```
yab config spark-thriftserver.default.port
yab config spark-thriftserver.default.username
yab config spark-thriftserver.default.password
```

Change to the Spark package directory. Start the beeline tool, passing the required hostname and port number. The actual hostname (not 'localhost') must be used.

```
cd <Spark package directory>
./bin/beeline

>beeline !connect jdbc:hive2://<hostname>:<ThriftServer port>
Connecting to jdbc:hive2://<hostname>:<ThriftServer port>
Enter username for jdbc:hive2://vmlwebs20t:22178: <ThriftServer username>
Enter password for jdbc:hive2://vmlwebs20t:22178: <ThriftServer password>
log4j:WARN No appenders could be found for logger (org.apache.hive.jdbc.Utils).
log4j:WARN Please initialize the log4j system properly.
log4j:WARN See http://logging.apache.org/log4j/1.2/faq.html#noconfig for more info.
```

```
Connected to: Spark SQL (version 2.4.5)
Driver: Hive JDBC (version 1.2.1.spark2)
Transaction isolation: TRANSACTION_REPEATABLE_READ
0: jdbc:hive2://<hostname>:<ThriftServer port>>
```

Alternatively, the above parameters can be passed in directly from the command line.

```
cd <Spark package directory>
./bin/beeline -u jdbc:hive2://<hostname>:<ThriftServer port> -n <ThriftServer username> -p
<ThriftServer password>

Connecting to jdbc:hive2://<hostname>:<ThriftServer port>
log4j:WARN No appenders could be found for logger (org.apache.hive.jdbc.Utils).
log4j:WARN Please initialize the log4j system properly.
log4j:WARN See http://logging.apache.org/log4j/1.2/faq.html#noconfig for more info.
Connected to: Spark SQL (version 2.4.5)
Driver: Hive JDBC (version 1.2.1.spark2)
Transaction isolation: TRANSACTION_REPEATABLE_READ
Beeline version 1.2.1.spark2 by Apache Hive
0: jdbc:hive2://<hostname>:<ThriftServer port>>
```

All Spark SQL commands must specify the name "qad_global_temp" of the Spark global temporary view, or no data will be shown. The global temporary view is essentially a virtual database that contains the datasets cached in the Query Service in the form of SQL views. Unfortunately, the global temporary view cannot be set as the default database for SQL commands, but must be specified explicitly in each one.

List the KPI datasets in the Spark global temporary view

The list shows the datasets stored in the Query Service, most of which are populated directly from the tables in the Cassandra data lake.

```
0: jdbc:hive2://vmlwebs20t:22178> show tables from qad_global_temp;
+------------------+-----------------------------------------------------+--------------+--+
|     database     |                      tableName                      | isTemporary  |
+------------------+-----------------------------------------------------+--------------+--+
| qad_global_temp  | kpi_1006542657_10usa                                | true         |
| qad_global_temp  | kpi_1006542657_11can                                | true         |
| qad_global_temp  | kpi_1006542657_12mex                                | true         |
| qad_global_temp  | kpi_1006542657_20fra                                | true         |
| qad_global_temp  | kpi_1006542657_21nl                                 | true         |
| qad_global_temp  | kpi_1006542657_22uk                                 | true         |
| qad_global_temp  | kpi_1006542657_23ger                                | true         |
| qad_global_temp  | kpi_1006542657_30chn                                | true         |
| qad_global_temp  | kpi_1006542657_31aus                                | true         |
| qad_global_temp  | kpi_1006542657_40brz                                | true         |
| qad_global_temp  | kpi_1006542657_80tst                                | true         |
| qad_global_temp  | kpi_1006542657_90trn                                | true         |
| qad_global_temp  | kpi_1006542657___74cfc248d7b221303172a6c252031069   | true         |
| qad_global_temp  | kpi_1006542657_qad                                  | true         |
...

335 rows selected (0.279 seconds)
0: jdbc:hive2://vmlwebs20t:22178>
```

Display the contents of a Spark dataset

The contents of any view (dataset) can also be displayed, although the data is often not very readable on a terminal console.

```
0: jdbc:hive2://vmlwebs20t:22178> select * from qad_global_temp.kpi_1006542657_10usa;
+-------------+--------------+-----------------------+-----------------------
+-----------------------+-----------------------+-----------------------
+------------------+-----------------------+-----------------------+-----------------------
+-----------------------+----------------------+--------------------------
+-----------------------+-------------------+------------------------------
+-------------------------------+-------------------------------+-------------------------------
```

```
+------------------------------+------------------------------+------------------------------
+------------------------------+------------------------------+------------------------------
+------------------------------+------------------------------+------------------------------
+------------------------------+------------------------------+------------------------------
+------------------------------+------------------------------+------------------------------
+------------------------------+----------------------+----------------------+--------------------
+--------------------+-----------------------+-----------------------+--------------------
+-----------------+--+
| domaincode  | sequencenbr  |  in_mstr__in_avg_date  |  in_mstr__in_avg_iss  |
in_mstr__in_avg_sls  | in_mstr__in_cnt_date  | in_mstr__in_domain  |  in_mstr__in_iss_date  |
in_mstr__in_part  | in_mstr__in_proj_use  |  in_mstr__in_qty_all  | in_mstr__in_qty_nonet  |
in_mstr__in_qty_oh   | in_mstr__in_qty_ord  |   in_mstr__in_qty_req    |  in_mstr__in_rec_date  |
in_mstr__in_site  | local_variables__local_var02  | local_variables__local_var03  |
local_variables__local_var04  | local_variables__local_var06  | local_variables__local_var08  |
local_variables__local_var09  | local_variables__local_var10  | local_variables__local_var11  |
local_variables__local_var12  | local_variables__local_var34  | local_variables__local_var36  |
local_variables__local_var38  | local_variables__local_var52  | local_variables__local_var54  |
local_variables__local_var55  | local_variables__local_var57  | local_variables__local_var90  |
local_variables__local_var92  | local_variables__local_var94  | local_variables__local_var96  |
pt_mstr__pt_abc_qty  |  pt_mstr__pt_desc1  | pt_mstr__pt_desc2  | pt_mstr__pt_group  |
pt_mstr__pt_part_type  | pt_mstr__pt_prod_line  | pt_mstr__pt_status  | pt_mstr__pt_um  |
+-------------+--------------+-----------------------+----------------------
+----------------------+----------------------+---------------------+-----------------------
+--------------------+-----------------------+-----------------------+-----------------------
+-----------------------+---------------------+--------------------------
+-----------------------+--------------------+-----------------------------
+------------------------------+------------------------------+------------------------------
+------------------------------+------------------------------+------------------------------
+------------------------------+------------------------------+------------------------------
+------------------------------+------------------------------+------------------------------
+------------------------------+------------------------------+------------------------------
+------------------------------+------------------------------+------------------------------
+------------------------------+----------------------+----------------------+--------------------
+--------------------+-----------------------+-----------------------+--------------------
+-----------------+--+
| 10USA       | 1            | 2016-02-16 16:00:00.0  | 0.523489932900000000  |
0.523489932900000000  | NULL                  | 10USA                | 2017-04-24 17:00:00.0  |
01010              | 0                    | 8.000000000000000000  | 0E-18                  |
9.000000000000000000  | 0E-18                | 1183.000000000000000000  | 2017-04-19 17:00:00.0
| 10-100            | A                             |                          |
M                             | false                         | 223217.740000000000000000     |
0E-18                         | 0E-18                         | 0E-18                         |
0E-18                         | 0E-18                         | 0E-18                         |
0E-18                         | 0E-18                         | 0                             |
0E-18                         | 0                             | 0E-18                         |
Standard                      | 24801.970000000000000000      | 0E-18                         |
0E-18                | Medical Ultrasound  | fff                | ULTRA              |
DEVICES                | 10                    | ACTIVE               | EA              |
| 10USA       | 2            | 2017-10-14 17:00:00.0  | 0E-18                | 0E-
18                 | NULL                  | 10USA               | 2017-10-15 17:00:00.0  |
01010              | 0                    | 0E-18                | 0E-18                  | 0E-
18                 | 0E-18                | 0E-18                   | 2017-10-15 17:00:00.0  |
dsite-18          | A                             |                          |
M                             | false                         | 0E-18                         |
0E-18                         | 0E-18                         | 0E-18                         |
0E-18                         | 0E-18                         | 0E-18                         |
0E-18                         | 0E-18                         | 0                             |
0E-18                         | 0                             | 0E-18                         |
Standard                      | 0E-18                         | 0E-18                         |
0E-18                | Medical Ultrasound  | fff                | ULTRA              |
DEVICES                | 10                    | ACTIVE               | EA              |
| 10USA       | 3            | NULL                 | 0E-18                | 0E-
18                 | NULL                  | 10USA               | NULL                   |
01010              | 0                    | 0E-18                | 0E-18                  | 0E-
18                 | 0E-18                | 0E-18                   | NULL                   |
dsite-8           | A                             |                          |
M                             | false                         | 0E-18                         |
0E-18                         | 0E-18                         | 0E-18                         |
0E-18                         | 0E-18                         | 0E-18                         |
0E-18                         | 0E-18                         | 0                             |
0E-18                         | 0                             | 0E-18                         |
Standard                      | 0E-18                         | 0E-18                         |
0E-18                | Medical Ultrasound  | fff                | ULTRA              |
DEVICES                | 10                    | ACTIVE               | EA              |
| 10USA       | 4            | NULL                 | 0E-18                | 0E-
18                 | NULL                  | 10USA               | NULL                   | 01010-
008         | 0                    | 0E-18                | 0E-18                  | 0E-
18                 | 0E-18                | 0E-18                   | NULL                   |
10-100            | b                             |
```

```
|                              | false                          | 0E-18                          |
0E-18                          | 0E-18                          | 0E-18                          |
0E-18                          | 0E-18                          | 0E-18                          |
0E-18                          | 0E-18                          | 0                              |
0E-18                          | 0                              | 0E-18                          |
Standard                       | 0E-18                          | 0E-18                          |
0E-18                | Verify Link to CC   |                          |
|                          |                          | ACTIVE               | EA               |
| 10USA       | 5            | NULL                    | 0E-18                 | 0E-
18                 | NULL                  | 10USA                | NULL                   | 01010-
014         | 0                     | 0E-18                | 0E-18                  | 0E-
18                 | 0E-18                 | 0E-18                    | NULL                   |
10-100            | a                              |
|                              | false                          | 0E-18                          |
0E-18                          | 0E-18                          | 0E-18                          |
0E-18                          | 0E-18                          | 0E-18                          |
0E-18                          | 0E-18                          | 0                              |
0E-18                          | 0                              | 0E-18                          |
Standard                       | 0E-18                          | 0E-18                          |
0E-18                | this is a testret   |                          |
|                          |                          |                    | EA               |
...
+-------------+-------------+-----------------------+----------------------
+-----------------------+----------------------+---------------------+-----------------------
+------------------+----------------------+----------------------+-----------------------
+----------------------+---------------------+-------------------------
+------------------------+------------------+-----------------------------
+-------------------------------+-------------------------------+-------------------------------
+-------------------------------+-------------------------------+-------------------------------
+-------------------------------+-------------------------------+-------------------------------
+-------------------------------+-------------------------------+-------------------------------
+-------------------------------+-------------------------------+-------------------------------
+-------------------------------+-------------------------------+-------------------------------
+-------------------------------+---------------------+--------------------+--------------------
+-------------------+-----------------------+-----------------------+--------------------
+-----------------+--+
745 rows selected (2.579 seconds)
0: jdbc:hive2://vmlwebs20t:22178>
```

To disconnect from the ThriftServer and exit beeline, use the '!quit' command.

```
0: jdbc:hive2://vmlwebs20t:22178> !quit
```

While inside beeline, use the '!help' command to get information about many other beeline commands.

```
0: jdbc:hive2://vmlwebs20t:22178> !help
```

Other Tools to Access Cassandra and Spark Data

In addition to the Cassandra and Spark command-line tools already described, it is possible to view the contents of Cassandra and/or Spark using third-party SQL client tools (examples: DBeaver, DbVisualizer, SQuirreL SQL) that support Cassandra or Spark JDBC drivers. The configuration details are specific to each tool and outside the scope of the current document, but for frequent users these tools may provide a more convenient, graphical, and usable alternative to the command-line utilities described here.

Enabling Debug Logging

In order to investigate the details of problems related to Action Centers and Query Service, it is sometimes necessary to enable debugging selectively for one or more of the components. Most of the common logging configuration changes to enable debugging can be done using YAB commands.

Any specialized logging configuration changes not supported by YAB properties and commands must be made manually in a text editor. If this is necessary, it should be done only in consultation with QAD Support personnel.

Tomcat

In order to investigate the details of problems related to Action Centers and Query Service, it is often necessary to enable debugging in the Tomcat instance supporting the Web UI. To accomplish this, the logging configuration file must be modified. To find the location of the Tomcat web app, query the YAB configuration.

```
yab config webapp.webshell.dir
```

The logging configuration file is named logback.xml, and is stored in the WEB-INF/config/ directory under the web app location. To obtain help about the relevant YAB properties and commands, run the following YAB command.

```
yab help logback.webshell
```

Debugging should be enabled selectively for relevant parts of the Tomcat environment, not globally for all Java classes. Depending on the problem to be debugged, the following Java packages and classes are usually the most important. These packages and classes are used as the 'NAMESPACE' entries referenced in the YAB help.

Component	Packages/Classes to Debug
Action Centers - Overall	com.qad.analytics.core, com.qad.analytics.core.mvc.controller.view, com.qad.analytics.core.mvc.controller.data, com.qad.analytics.core.lps, com.qad.analytics.core.service, com.qad.analytics.core.util
Action Centers - Financials with FRW (deprecated)	com.qad.analytics.financials, com.qad.analytics.financials.mvc.controller.data, com.qad.analytics.financials.service, com.qad.analytics.financials.util
Query Service - Spark processing	com.qad.qracore.service.impl.spark
Query Service - Cassandra processing	com.qad.qracore.service.impl.BrowseDefinitionServiceImpl, com.qad.qracore.service.impl.CassandraCopyServiceContext, com.qad.qracore.service.impl.CassandraCopyServiceImpl, com.qad.qracore.service.impl.CassandraCopyTask, com.qad.qracore.service.impl.CassandraCopyTaskFactory

For example, to enable debug-level logging for the *com.qad.analytics.core.service* package, set the following YAB property.

```
logback.webshell.loglevel.com.qad.analytics.core.service=debug
```

⚠ Enabling debug-level logging can cause the Tomcat log file size to expand quickly. It should be used only selectively and for short periods of time in production environments.

After the desired YAB properties have been set, run the following YAB command to update the logging settings.

```
yab logback-webshell-update
```

Once the changes are saved, they take effect within a minute or so with no need to restart the Tomcat instance.

Logi Platform Services

When investigating problems in the display of visuals inside the Action Centers supported by Logi Platform Services, expanded panels, or the KPI screen, it is sometimes helpful to enable debugging for the LogiPS Application Service and /or Data Service. To do this, set the following YAB properties.

```
logi-platform-services.default.service.application.loglevel=debug
logi-platform-services.default.service.data.loglevel=debug
```

After these properties have been set, restart Logi Platform Services. There is no need to restart Tomcat.

Logi Info

When investigating problems in the display of visuals or grids inside the Action Centers supported by Logi Info, expanded panels, or the KPI screen, it is sometimes helpful to enable debugging for the Logi plugin classes specific to Action Centers. Most of the common logging configuration changes to enable debugging can be done using YAB commands. To find the location of the Tomcat webapp, query the YAB configuration.

```
yab config webapp.analytics-logi.dir
```

The logging configuration file is named log4j.properties, and is stored in the WEB-INF/classes/ directory under the web app location. To obtain help about the relevant YAB properties and commands, run the following YAB command.

```
yab help log4j.analytics-logi
```

To enable debugging for the QAD Logi plugin classes, set the following YAB property to change the root log level from FATAL to INFO.

```
log4j.analytics-logi.loglevel=info
```


Do not set the Logi log level to DEBUG, as this causes many internal Logi log messages to be written that are unlikely to be helpful in Action Center problem diagnosis.

After the desired YAB properties have been set, run the following YAB command to update the logging settings.

```
yab log4j-analytics-logi-update
```

The Tomcat instance must be restarted for the new log level to take effect.

Checking the Financial Daemons

The Financial Report Writer (FRW) feature that supports Action Centers relies on background processes called daemons to continuously update the FRW KPI data as General Ledger entries are created. If some of these daemons are not running, the FRW KPIs and Action Centers may not reflect all current financial activity. In particular, the History Daemon and Cube Daemon are required.

The daemons are not automatically started when the environment is started using YAB. To check if the daemons are running, run the following YAB command.

```
yab daemon-status

                    daemon-status (13 tasks)                          [APPLY]
-------------------------------------------------------------------------------
1/13  daemon-xmldaemon-status                          STOPPED (1.591 s)
2/13  daemon-balancedaemon-status                      STOPPED (0.629 s)
3/13  daemon-budgetdaemon-status                       STOPPED (0.619 s)
4/13  daemon-crosscompanydaemon-status                 STOPPED (0.573 s)
5/13  daemon-eventdaemon-status                        STOPPED (0.627 s)
6/13  daemon-historydaemon-status                      STOPPED (0.621 s)
7/13  daemon-replicationdaemon-status                  STOPPED (0.621 s)
8/13  daemon-scandaemon-status                         STOPPED (0.621 s)
9/13  daemon-timeoutdaemon-status                      STOPPED (0.619 s)
10/13 daemon-cubedaemon-status                         STOPPED (0.619 s)
11/13 daemon-reportdaemon-status                       STOPPED (0.622 s)
12/13 daemon-batchdaemon-status                        STOPPED (0.618 s)
13/13 daemon-status                                    OK (1.933 s)
-------------------------------------------------------------------------------
```

To start the history and cube daemons, run the following YAB command.

```
yab daemon-historydaemon-start daemon-cubedaemon-start
```

```
   daemon-historydaemon-start daemon-cubedaemon-start (2 tasks)        [APPLY]
--------------------------------------------------------------------------------
1/2 daemon-historydaemon-start                                 STARTED (18.434 s)
2/2 daemon-cubedaemon-start                                    STARTED (6.833 s)
--------------------------------------------------------------------------------
```

The daemons also can be monitored, started, and stopped from screens in the .NET UI. Daemon setup is described in the *QAD System Administration User Guide*.

Action Center Disaster Recovery

Introduction

In a disaster recovery scenario where data stored in the Adaptive UX databases are lost or damaged and cannot be restored from current replicas for any reason, it is possible that some changes made to KPIs and Action Centers since the time of the most recent database backup will be lost, and must be manually re-entered once service is restored. This risk exists because of differences in the built-in capabilities of the various databases that support the Action Centers.

- **OpenEdge databases**: Contains the KPIs and browse definitions that define the data required for the visuals and Action Centers, as well as the dashboard resource identifiers and access permissions for each Action Center displayed on the Web UI menu.
- **Logi Platform Services H2 product database**: Contains the definition of all visuals, and the contents and layout of all Action Centers.
- **Cassandra data lake**: Contains current and historical snapshots of the KPI datasets extracted from the operational databases, mainly browse result sets.

This section summarizes the disaster recovery procedures required or recommended for Action Center data. However, it does not cover the specific YAB commands that implement database backups and restores. For this information, see the YAB Configuration and Administration Guide or use the YAB help command.

OpenEdge Databases

The OpenEdge databases are backed up regularly and frequently. In addition, they are protected in case of disaster by roll-forward logging and after-imaging, which allows transactions committed since the previous backup to be rolled forward into a backup copy of the database to bring it up to date, This is most important to protect the business and transaction data outside of analytics, but also applies to the KPI and Action Center definitions stored in the same qaddb database. Thus, all KPI and Action Center definitions added/modified/deleted since the disaster can be restored without data loss. However, these definitions may not be in sync with Logi Platform Services, as described below.

Logi Platform Services H2 Product Database

The Logi Platform Services product database is backed up at the same time as the OpenEdge databases, and all the backups will therefore be in sync. When the most recent backups for OpenEdge and Logi are restored following a disaster, their contents will agree. In addition, no in-flight H2 transactions will be lost, as the H2 commits are written to disk immediately, not buffered in memory and flushed to disk later. However, unlike the OpenEdge databases, the H2 product database within Logi Platform Services does not have after-imaging or roll-forward capability. While databases transactions since the last backup can be applied automatically to the restored OpenEdge databases to make them current as of the time of the service outage, the same cannot be done for the Logi product database. This database contains only metadata, such as visual and dashboard definitions, which are relatively static and not updated as frequently as the business data. Roll-forward capability is not nearly as important as with other Adaptive UX data. However, this difference implies that following a disaster and roll-forward recovery, changes made to the Logi product database since the most recent backup may be lost and recoverable only through manual re-entry. This section describes in more specific terms the information that could be lost.

New or Changed Visuals Published to the Gallery

The most common and frequent kind of change that affects Action Centers is the creation and modification of charts and other visuals. While visuals may be created by individual Adaptive UX users for their own use, as a best practice visuals should be published to the gallery so that other Action Center users can benefit from them.

After a disaster, Action Center users should be notified that any visuals added or modified since the restored database backup must be re-created and re-published manually. If the backup was taken recently and the duration of the service outage was short, users may be able to recall and re-apply their changes without a large effort.

Often, visuals are created or modified in combination with changes to the KPI definition with which those visuals are associated. The last modified date-time of each KPI is stored in the OpenEdge database. Following roll-forward actions to fully restore the OpenEdge database, it may be helpful to identify those KPIs that were modified since the restored backup using a database query. Users who created or use those KPIs in particular could then be requested to check the associated visuals for currency, as an additional reminder.

The following ABL query can be run in the Progress Editor of the NetUI to identify those KPIs modified since the backup.

```
for each KpiMetadata no-lock where LastModifiedDateTime > datetime-tz(<date-time of restored
backup>):
display KpiName LastModifiedUser LastModifiedDateTime.
```

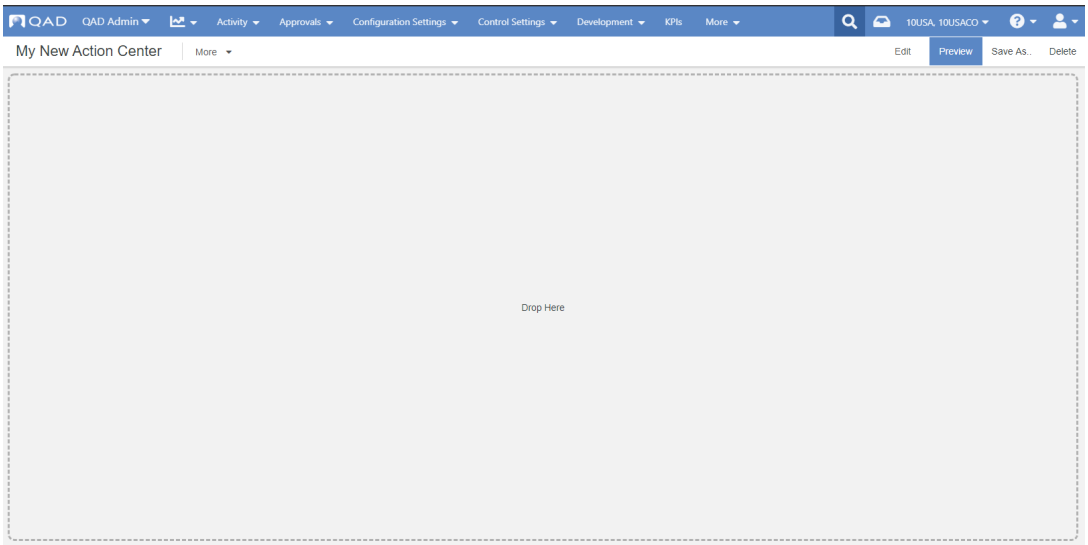
Alternatively, the following OpenEdge SQL query will also work in the Progress Editor.

```
select KpiName, LastModifiedUser, LastModifiedDateTime from KpiMetadata
where LastModifiedDateTime > datetime-tz(<date-time of restored backup>)
```

The date-time literal in the above expressions does not have to be quoted, but must be in the correct OpenEdge date-format for the database (for example, '05/31/2020' for US databases in 'MDY' format).

New Action Centers

If an Action Center was created since the last database backup prior to the service outage, following database restore and roll-forward actions it will be present on the Web UI menu, but the contents will not be present in Logi Platform Services. In this case, displaying the Action Center from the Web UI will not raise an error, but simply display an empty dashboard.



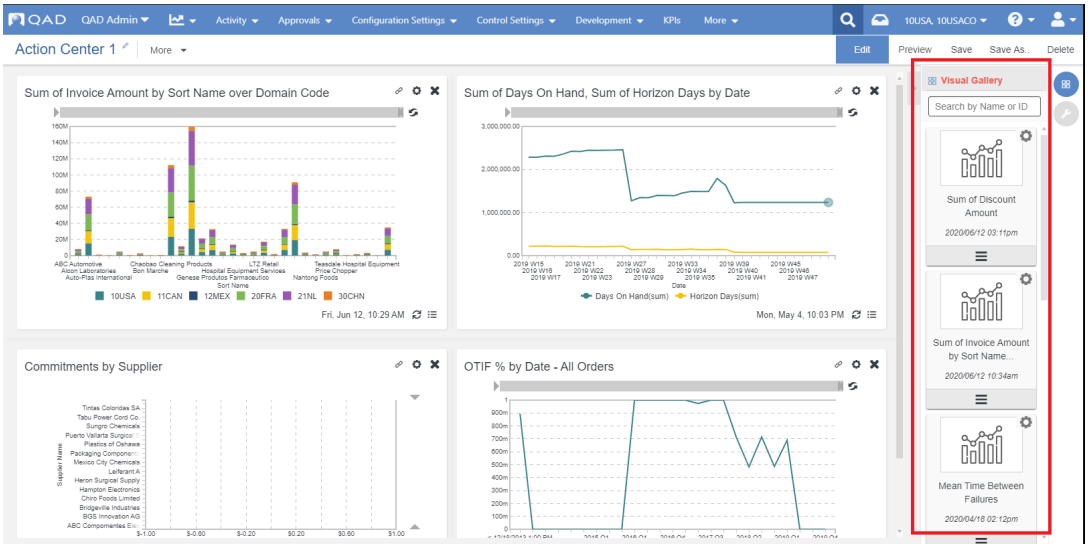
The user can then re-create the Action Center using the same procedure as when it was created originally.

Modified Action Center Contents and Layouts

For an Action Center modified since the last database backup prior to the service outage, the recovery procedure is straightforward. The Action Center will display correctly from the Web UI menu with its old contents, and the user can re-apply the changes that were lost in the same way as the changes were made originally.

Deleted Visuals

Visuals that were deleted from Logi Platform Services since the last database backup prior to the service outage will have to be deleted again, once OpenEdge database restore and roll-forward actions are complete. To do this, a user with dashboard sharing permissions can display any Action Center that he/she is allowed to edit, press the Edit button, and press the Visual Gallery button  . The gallery will be displayed in a sidebar on the right side of the window.



Unwanted visuals can be selected with the help of the search bar at the top. Selected visuals are deleted from the gallery by pressing the cog icon and selecting the Delete command. However, any visual to be deleted must first be removed from Action Centers where it is being used, or the delete action will not be allowed.

Deleted Action Centers

Action Centers that were deleted from Logi Platform Services since the last database backup prior to the service outage wil l be absent from the Web UI menu, once OpenEdge database restore and roll-forward actions are complete. Because they are no longer known and cannot be seen by the Web UI, they can be deleted only through Logi Platform Services directly. While this can be accomplished using internal Logi APIs, it is outside the scope of the present document to describe these details. It is recommended to contact QAD Service Delivery for assistance in this case (AB-26845).

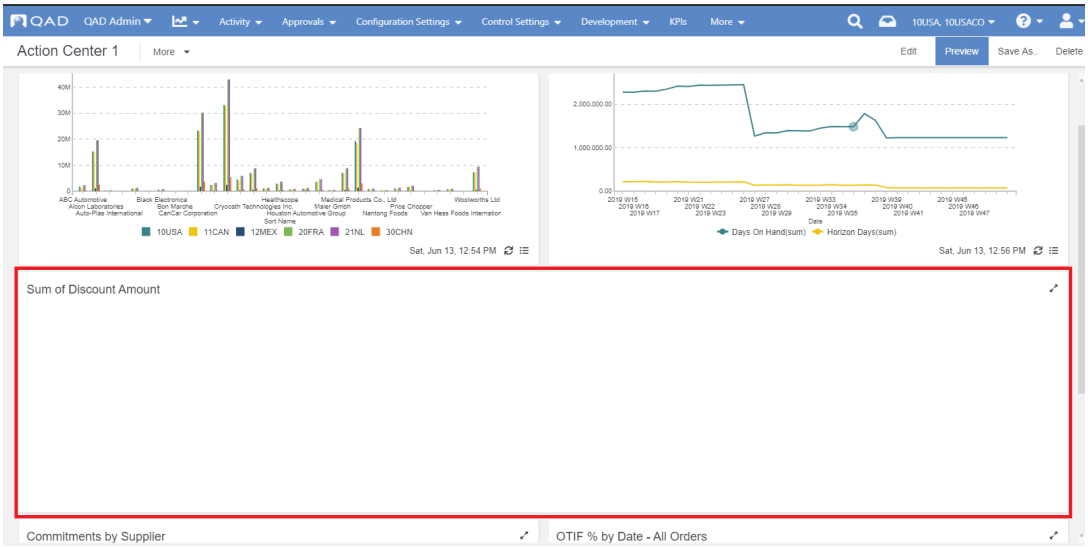
Because deleted Action Centers no longer exist in the Web UI at all, they cannot be accessed by any QAD user and therefore have no impact on other Adaptive UX functions.

What If Action Centers in Logi Platform Services are Out of Sync with OpenEdge Data?

In most cases where lost Action Center contents cannot be re-created manually or will take some time to re-create, the out-of-date Action Centers and visuals should display without error with the old visuals and layouts. However, there are several exceptions where special attention is needed.

KPI Fields Removed or Renamed

One possibility is that a KPI field that is being used in visuals was deleted or renamed since the last restored backup. Standard Action Center validation prevents any field used in a visual from being deleted or renamed, and in this case the field would have had to be removed from all visuals before the delete or rename action. However, following disaster recovery the current OpenEdge database might be running with an older Logi Platform Services product database that still references the old field. In this kind of case, an Action Center panel containing the obsolete visual would display as blank.



In addition, the console log file for the tomcat-webui web server, normally catalina.out, would contain an error referencing the deleted/renamed field. In the following example, the 'Discount Amount' field that is used on the above visual 'Sum of Discount Amount' has been deleted.

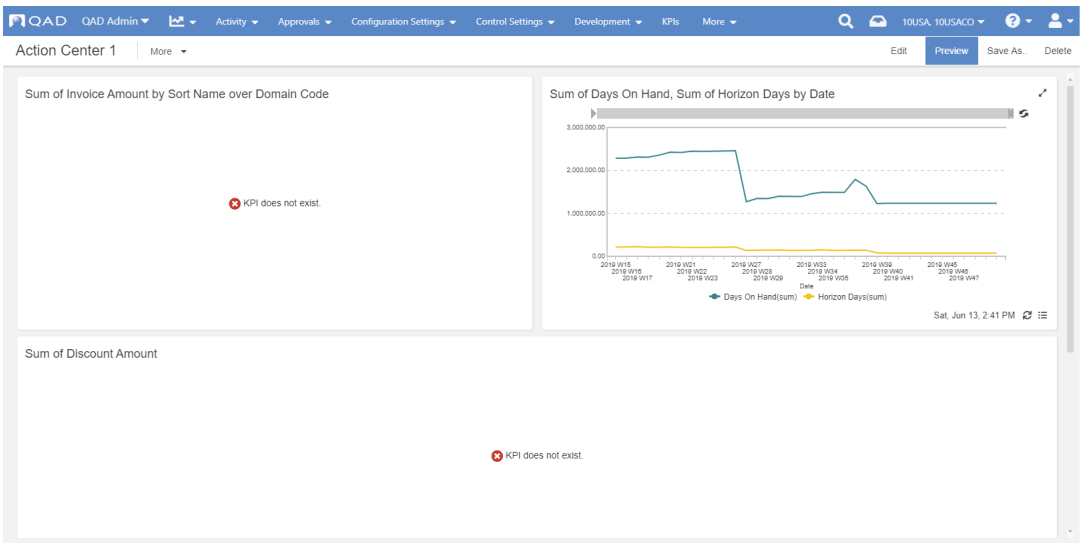
```
ERROR[HiveServer2-Background-Pool: Thread-1271] o.a.s.s.h.t.SparkExecuteStatementOperation:91
Error executing query, currentState RUNNING,
org.apache.spark.sql.AnalysisException: cannot resolve '`Query.Discount_Amount`' given input
columns: [Query.idh_hist__idh_qty_ord, Query.ih_hist__ih_ship, Query.ih_hist__ih_cust, Query.
idh_hist__idh_list_pr, Query.idh_hist__idh_qty_inv, Query.ih_hist__ih_bill, Query.
idh_hist__idh_due_date, Query.ad_mstr__ad_sort, Query.ih_hist__ih_nbr, Query.
local_variables__local_var03, Query.idh_hist__idh_part, Query.domainCode, Query.
idh_hist__idh_site, Query.pt_mstr__pt_desc1]; line 1 pos 59;
'GlobalLimit 201
+- 'LocalLimit 201
   +- 'Aggregate [count(1) AS Discount_Amount_count#14002L, 'COUNT('Query.Discount_Amount) AS
Discount_Amount_distinct#14003, 'MIN('Query.Discount_Amount) AS Discount_Amount_min#14004, 'MAX
('Query.Discount_Amount) AS Discount_Amount_max#14005, 'AVG(cast('Query.Discount_Amount as decimal
(28,5))) AS Discount_Amount_avg#14006, 'SUM(cast('Query.Discount_Amount as decimal(28,5))) AS
Discount_Amount_sum#14007]
      +- Filter upper(domainCode#8120) IN (10USA,10USA,11CAN,11CAN,12MEX,20FRA,21NL,21NL,22UK,
22UK,23GER,30CHN,31AUS,31AUS,40BRZ,80TST,90TRN,QAD)
         +- SubqueryAlias `Query`
            +- Project [ad_mstr__ad_sort#8122, to_utc_timestamp(idh_hist__idh_due_date#8124,
America/Los_Angeles) AS idh_hist__idh_due_date#14001, idh_hist__idh_list_pr#8126,
idh_hist__idh_part#8127, idh_hist__idh_qty_inv#8129, idh_hist__idh_qty_ord#8130,
idh_hist__idh_site#8131, ih_hist__ih_bill#8134, ih_hist__ih_cust#8137, ih_hist__ih_nbr#8141,
ih_hist__ih_ship#8143, local_variables__local_var03#8148, pt_mstr__pt_desc1#8150, domainCode#8120]
...
    at org.apache.spark.sql.catalyst.analysis.package$AnalysisErrorAt.failAnalysis(package.scala:
42)
    at org.apache.spark.sql.catalyst.analysis.
CheckAnalysis$$anonfun$checkAnalysis$1$$anonfun$apply$3.applyOrElse(CheckAnalysis.scala:111)
    at org.apache.spark.sql.catalyst.analysis.
CheckAnalysis$$anonfun$checkAnalysis$1$$anonfun$apply$3.applyOrElse(CheckAnalysis.scala:108)
    at org.apache.spark.sql.catalyst.trees.TreeNode$$anonfun$transformUp$1.apply(TreeNode.scala:
280)
    at org.apache.spark.sql.catalyst.trees.TreeNode$$anonfun$transformUp$1.apply(TreeNode.scala:
280)
    at org.apache.spark.sql.catalyst.trees.CurrentOrigin$.withOrigin(TreeNode.scala:69)
    at org.apache.spark.sql.catalyst.trees.TreeNode.transformUp(TreeNode.scala:279)
    at org.apache.spark.sql.catalyst.trees.TreeNode$$anonfun$3.apply(TreeNode.scala:277)
    at org.apache.spark.sql.catalyst.trees.TreeNode$$anonfun$3.apply(TreeNode.scala:277)
    at org.apache.spark.sql.catalyst.trees.TreeNode.
org$apache$spark$sql$catalyst$trees$TreeNode$$mapChild$2(TreeNode.scala:297)
    at org.apache.spark.sql.catalyst.trees.TreeNode$$anonfun$4$$anonfun$apply$13.apply(TreeNode.
scala:356)
    at scala.collection.TraversableLike$$anonfun$map$1.apply(TraversableLike.scala:234)
    at scala.collection.TraversableLike$$anonfun$map$1.apply(TraversableLike.scala:234)
    at scala.collection.mutable.ResizableArray$class.foreach(ResizableArray.scala:59)
    at scala.collection.mutable.ArrayBuffer.foreach(ArrayBuffer.scala:48)
    at scala.collection.TraversableLike$class.map(TraversableLike.scala:234)
    at scala.collection.AbstractTraversable.map(Traversable.scala:104)
...
    at org.apache.spark.sql.SparkSession.sql(SparkSession.scala:642)
```

```
    at org.apache.spark.sql.SQLContext.sql(SQLContext.scala:694)
    at org.apache.spark.sql.hive.thriftserver.SparkExecuteStatementOperation.
org$apache$spark$sql$hive$thriftserver$SparkExecuteStatementOperation$$execute
(SparkExecuteStatementOperation.scala:232)
    at org.apache.spark.sql.hive.thriftserver.SparkExecuteStatementOperation$$anon$1$$anon$2.run
(SparkExecuteStatementOperation.scala:175)
    at org.apache.spark.sql.hive.thriftserver.SparkExecuteStatementOperation$$anon$1$$anon$2.run
(SparkExecuteStatementOperation.scala:171)
    at java.security.AccessController.doPrivileged(Native Method)
    at javax.security.auth.Subject.doAs(Subject.java:422)
    at org.apache.hadoop.security.UserGroupInformation.doAs(UserGroupInformation.java:1893)
    at org.apache.spark.sql.hive.thriftserver.SparkExecuteStatementOperation$$anon$1.run
(SparkExecuteStatementOperation.scala:185)
    at java.util.concurrent.Executors$RunnableAdapter.call(Executors.java:511)
    at java.util.concurrent.FutureTask.run(FutureTask.java:266)
    at java.util.concurrent.ThreadPoolExecutor.runWorker(ThreadPoolExecutor.java:1149)
    at java.util.concurrent.ThreadPoolExecutor$Worker.run(ThreadPoolExecutor.java:624)
    at java.lang.Thread.run(Thread.java:748)
```

In this case, the visual must be modified or re-created without referencing the KPI field that is no longer valid. The Action Center containing the visual must then be modified to include the replacement visual.

Browse Definition Modified

A similar problem occurs when the definition of the source browse associated with a KPI was modified since the last restored backup to remove fields that were being used by the KPI. In this case, following disaster recovery the current OpenEdge database and browse definitions might be running with an older Logi Platform Services product database based on the old browse. In this kind of case, the KPI referencing the non-existent browse field would be broken. An Action Center panel containing visuals associated with that KPI would display with errors.



In addition, the console log file for the tomcat-webui web server would contain one or more errors referencing the KPI.

```
ERROR[http-bio-22011-exec-2] c.q.a.c.l.LogiPlatformWidgetSetupService:154
getLogiDashboardPanelSetupInfo(): error creating data visualization panel info for KPI code:
9be65599-692e-f782-6214-94e3c0caec26
java.lang.NullPointerException: null
    at com.qad.analytics.core.lps.DataLakeFieldService.lambda$2(DataLakeFieldService.java:60)
    at java.util.stream.ReferencePipeline$3$1.accept(ReferencePipeline.java:193)
    at java.util.stream.ReferencePipeline$2$1.accept(ReferencePipeline.java:175)
    at java.util.Iterator.forEachRemaining(Iterator.java:116)
    at java.util.Spliterators$IteratorSpliterator.forEachRemaining(Spliterators.java:1801)
    at java.util.stream.AbstractPipeline.copyInto(AbstractPipeline.java:482)
    at java.util.stream.AbstractPipeline.wrapAndCopyInto(AbstractPipeline.java:472)
    at java.util.stream.StreamSpliterators$WrappingSpliterator.forEachRemaining
(StreamSpliterators.java:312)
    at java.util.stream.Streams$ConcatSpliterator.forEachRemaining(Streams.java:742)
    at java.util.stream.AbstractPipeline.copyInto(AbstractPipeline.java:482)
    at java.util.stream.AbstractPipeline.wrapAndCopyInto(AbstractPipeline.java:472)
    at java.util.stream.ReduceOps$ReduceOp.evaluateSequential(ReduceOps.java:708)
    at java.util.stream.AbstractPipeline.evaluate(AbstractPipeline.java:234)
    at java.util.stream.ReferencePipeline.collect(ReferencePipeline.java:499)
```

```
    at com.qad.analytics.core.lps.LogiPlatformWidgetSetupInfoService.fillDataLakeInfo
(LogiPlatformWidgetSetupInfoService.java:195)
    at com.qad.analytics.core.lps.LogiPlatformWidgetSetupInfoService.
setUpLogiKpiBasedWidgetSetupInfo(LogiPlatformWidgetSetupInfoService.java:178)
    at com.qad.analytics.core.lps.LogiPlatformWidgetSetupInfoService.
getLogiDashboardPanelSetupInfo(LogiPlatformWidgetSetupInfoService.java:148)
    at com.qad.analytics.core.lps.LogiPlatformWidgetSetupService.getLogiDashboardPanelSetupInfo
(LogiPlatformWidgetSetupService.java:152)
...
    at com.qad.analytics.core.lps.LogiPlatformWidgetSetupService.getLogiDashboardPanelSetupInfos
(LogiPlatformWidgetSetupService.java:142)
    at com.qad.analytics.core.lps.mvc.controller.data.WidgetSetupController.
setupDashboardPanels_aroundBody14(WidgetSetupController.java:98)
    at com.qad.analytics.core.lps.mvc.controller.data.WidgetSetupController$AjcClosure15.run
(WidgetSetupController.java:1)
    at org.aspectj.runtime.reflect.JoinPointImpl.proceed(JoinPointImpl.java:149)
    at com.qad.qracore.mvc.interceptor.BEExtensionInterceptorImpl.executionAround
(BEExtensionInterceptorImpl.java:143)
    at com.qad.webshell.aspects.BEExtensionAspect.executionAround(BEExtensionAspect.java:28)
    at com.qad.analytics.core.lps.mvc.controller.data.WidgetSetupController.
setupDashboardPanels_aroundBody16(WidgetSetupController.java:96)
    at com.qad.analytics.core.lps.mvc.controller.data.WidgetSetupController$AjcClosure17.run
(WidgetSetupController.java:1)
...
```

In this case, multiple steps are generally required to recover.

1. Any visuals associated with the KPI that reference the deleted field must be either deleted or modified to remove the obsolete field reference.
2. The KPI definition must be re-configured in the KPI screen for the updated browse by pressing the Configure button to display the browse, then pressing OK. Once the KPI definition is correct, the KPI can be saved.
3. If necessary, existing visuals must be modified or new ones created for the KPI that reference the correct data.
4. Any Action Center containing visuals that were removed in an earlier step should be modified to include replacement visuals.

Cassandra Data Lake

The Cassandra database is used for several purposes related to Action Centers. It contains multiple keyspaces that must be considered separately.

browses Keyspace

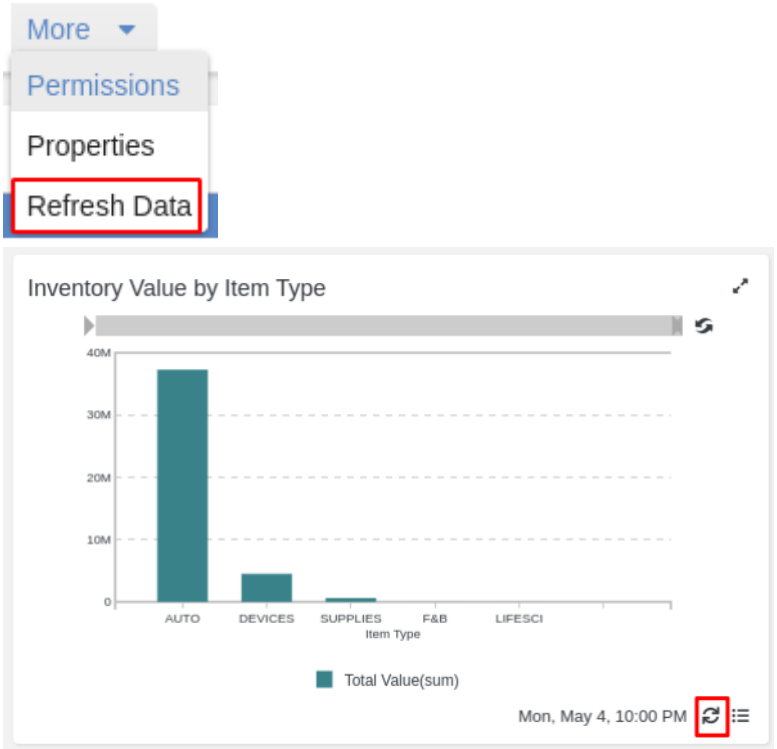
In the Cassandra data lake, the **browses** keyspace is used to hold current KPI data as of the latest refresh, whether scheduled or manual. The datasets may be accessed on line from Action Centers displayed in the Web UI.

The detailed KPI and browse data stored in the browses keyspace are normally not backed up, as they are created from the current contents of the OpenEdge databases. In case of disaster, the contents of this Cassandra keyspace can be re-created using the standard scheduled refresh and cache warming processes, once those databases have been recovered following the service outage. Hence, there is virtually no risk that the KPI datasets directly consumed by and displayed in Action Center visuals would be permanently lost.

In case of disaster, the OpenEdge databases should first be restored using normal procedures. Once this is done and Cassandra is running again, the browses keyspace will be re-populated automatically at tomcat-webui startup, assuming the cache warming is enabled, and Action Center will display normally with current data from the OpenEdge databases.

If cache warming is not enabled, the KPI data must be re-created using one of the following options.

- **Scheduled refresh**: For those KPIs that are configured to refresh daily, weekly, or monthly, their Cassandra data will be populated automatically at the next scheduled refresh. In this case, however, Action Center users would have to wait to see current data for these KPIs.
- **Manual refresh**: Web UI users of Action Centers who are authorized to manually refresh KPIs that have been configured with manual refresh enabled can do so from the Action Center screens. Under the 'More' menu on the Action Center toolbar, there is a 'Refresh Data' command that will re-populate all Cassandra tables for the KPIs used in that Action Center. Data refreshes for particular dashboard panels can also be triggered by pressing the refresh icon in the lower right-hand corner of each panel.



historical_kpi Keyspace

The **historical_kpi** keyspace is used to hold KPI data from historical snapshots taken at scheduled date-times, which may extend years into the past. Unlike the browses keyspace, its contents cannot be re-created from the current OpenEdge databases. In this case, Cassandra is the system of record for the data and should be backed up at the same time the OpenEdge databases are backed up each day. This can be accomplished by ensuring that the historical_kpi keyspace is absent from the blacklist of Cassandra keyspaces excluded from the YAB environment-backup operation, listed in the property cassandra.default.node.backup.blacklist. YAB backs up the designated keyspaces using the Cassandra 'nodetool snapshot' command.

If there is a service outage while data is being written into Cassandra, no in-flight transactions are lost. Cassandra first writes all changes to a commit log before treating them as successful. If the system crashes before those writes have been saved in the persistent database tables, the commit log is automatically replayed when the node is restarted to read them into in-memory 'memtables', from which they are flushed to disk.

Once the Cassandra environment has been restored following a disaster, the historical_kpi keyspace will contain all data as of the time of the service outage. However, historical snapshots that were in process and interrupted at the time of the service outage may contain only a portion of the required data, and therefore may not be usable. In this case, the snapshot failure will be detected at tomcat-webui startup, and a new historical snapshot will be taken automatically to replace each one that failed. If new activity has been processed by Adaptive UX since the restoration of service but before replacement snapshots are created, it is possible that the contents of the new snapshots will be different than the contents that would have been included in the respective failed snapshots. However, in most disaster recovery scenarios this window of time will be short, reducing the risk that the new KPI snapshot will be inaccurate by a significant amount.

Global Order Management Distribution Processing

Global Order Management Distribution Processing enables Available-To-Promise / Enterprise Materials Transfer (ATP /EMT) Visibility and EMT Tracking across domains of multiple instances of Enterprise Edition, where you have a central instance that can have visibility into all the others.

Please contact QAD Services to implement this capability in your system. *QAD Internal Link: Distributed Processing*.