Oracle Adaptive Intelligent Apps for CX
Implementing Adaptive Intelligent Commerce and Marketing

February 2020

Part Number: E89331-25

Copyright © 2017, 2020, Oracle and/or its affiliates.

Authors: Adaptive Intelligent Apps Information Development

This software and related documentation are provided under a license agreement containing restrictions on use and disclosure and are protected by intellectual property laws. Except as expressly permitted in your license agreement or allowed by law, you may not use, copy, reproduce, translate, broadcast, modify, license, transmit, distribute, exhibit, perform, publish, or display in any part, in any form, or by any means. Reverse engineering, disassembly, or decompilation of this software, unless required by law for interoperability, is prohibited.

The information contained herein is subject to change without notice and is not warranted to be error-free. If you find any errors, please report them to us in writing.

If this is software or related documentation that is delivered to the U.S. Government or anyone licensing it on behalf of the U.S. Government, then the following notice is applicable:

U.S. GOVERNMENT END USERS: Oracle programs (including any operating system, integrated software, any programs embedded, installed or activated on delivered hardware, and modifications of such programs) and Oracle computer documentation or other Oracle data delivered to or accessed by U.S. Government end users are "commercial computer software" or "commercial computer software documentation" pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, the use, reproduction, duplication, release, display, disclosure, modification, preparation of derivative works, and/or adaptation of i) Oracle programs (including any operating system, integrated software, any programs embedded, installed or activated on delivered hardware, and modifications of such programs), ii) Oracle computer documentation and/or iii) other Oracle data, is subject to the rights and limitations specified in the license contained in the applicable contract. The terms governing the U.S. Government's use of Oracle cloud services are defined by the applicable contract for such services. No other rights are granted to the U.S. Government.

This software or hardware is developed for general use in a variety of information management applications. It is not developed or intended for use in any inherently dangerous applications, including applications that may create a risk of personal injury. If you use this software or hardware in dangerous applications, then you shall be responsible to take all appropriate fail-safe, backup, redundancy, and other measures to ensure its safe use. Oracle Corporation and its affiliates disclaim any liability for any damages caused by use of this software or hardware in dangerous applications.

Oracle and Java are registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners.

Intel and Intel Inside are trademarks or registered trademarks of Intel Corporation. All SPARC trademarks are used under license and are trademarks or registered trademarks of SPARC International, Inc. AMD, Epyc, and the AMD logo are trademarks or registered trademarks of Advanced Micro Devices. UNIX is a registered trademark of The Open Group.

This software or hardware and documentation may provide access to or information about content, products, and services from third parties. Oracle Corporation and its affiliates are not responsible for and expressly disclaim all warranties of any kind with respect to third-party content, products, and services unless otherwise set forth in an applicable agreement between you and Oracle. Oracle Corporation and its affiliates will not be responsible for any loss, costs, or damages incurred due to your access to or use of third-party content, products, or services, except as set forth in an applicable agreement between you and Oracle.

The business names used in this documentation are fictitious, and are not intended to identify any real companies currently or previously in existence.
## Contents

<table>
<thead>
<tr>
<th>Preface</th>
<th>i</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>1 Get Started</strong></td>
<td>1</td>
</tr>
<tr>
<td>About Adaptive Intelligent Commerce and Marketing</td>
<td>1</td>
</tr>
<tr>
<td>About Oracle Cloud and Oracle Adaptive Intelligent Apps</td>
<td>2</td>
</tr>
<tr>
<td>Version Compatibility</td>
<td>3</td>
</tr>
<tr>
<td>Roles, User Administration, and Data Security</td>
<td>3</td>
</tr>
<tr>
<td>Data Sources</td>
<td>4</td>
</tr>
<tr>
<td>Privacy Regulations and Data Protection</td>
<td>5</td>
</tr>
<tr>
<td><strong>2 Import Commerce Data</strong></td>
<td>11</td>
</tr>
<tr>
<td>About Commerce Integrations</td>
<td>11</td>
</tr>
<tr>
<td>Import Data from Oracle Commerce Cloud</td>
<td>12</td>
</tr>
<tr>
<td>Import Data from Oracle Commerce Platform</td>
<td>13</td>
</tr>
<tr>
<td>Import Data from Other Commerce Applications</td>
<td>20</td>
</tr>
<tr>
<td>View Data Ingestion Status</td>
<td>33</td>
</tr>
<tr>
<td>View Clickstream Status</td>
<td>35</td>
</tr>
<tr>
<td><strong>3 Import Marketing Data</strong></td>
<td>37</td>
</tr>
<tr>
<td>About Marketing Integrations</td>
<td>37</td>
</tr>
<tr>
<td>Import Data from Oracle Responsys</td>
<td>37</td>
</tr>
<tr>
<td>Export Data from Oracle Responsys</td>
<td>40</td>
</tr>
<tr>
<td><strong>4 Configure Oracle BlueKai</strong></td>
<td>47</td>
</tr>
<tr>
<td>Connect Audiences for Oracle BlueKai</td>
<td>47</td>
</tr>
<tr>
<td><strong>5 Configure Web Widgets</strong></td>
<td>49</td>
</tr>
<tr>
<td>Web Widget Templates</td>
<td>49</td>
</tr>
<tr>
<td>Configure Widgets for Oracle Commerce Cloud</td>
<td>51</td>
</tr>
<tr>
<td>Configure Widgets for Oracle Commerce Platform</td>
<td>55</td>
</tr>
</tbody>
</table>
Configure Widgets for Other Commerce Applications 57
Override Default Titles for Recommendation Widgets 60
Clickstream Events 60

## 6 Configure Email

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>About Email Templates</td>
<td>63</td>
</tr>
<tr>
<td>Configure Email Widgets</td>
<td>64</td>
</tr>
<tr>
<td>Email Template Code</td>
<td>65</td>
</tr>
<tr>
<td>Configure Oracle Responsys</td>
<td>67</td>
</tr>
</tbody>
</table>

## 7 Troubleshooting

<table>
<thead>
<tr>
<th>Issue</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Why are promotions missing on the Insights page?</td>
<td>71</td>
</tr>
<tr>
<td>Why isn't my price or inventory data being updated?</td>
<td>71</td>
</tr>
<tr>
<td>Why aren't order or catalog changes reflected on the data warehouse server?</td>
<td>71</td>
</tr>
</tbody>
</table>
Preface

This preface introduces information sources that can help you use the application and this guide.

Using Oracle Applications
To find guides for Oracle Applications, go to the Oracle Help Center.

Documentation Accessibility
For information about Oracle's commitment to accessibility, visit the Oracle Accessibility Program website.

Contacting Oracle
Customers can access electronic support through Oracle Support. For information, visit My Oracle Support or visit Accessible Oracle Support if you are hearing impaired.

Please give us feedback about Oracle Applications Help and guides. You can send an e-mail to: oracle_aiapps_doc_feedback_grp@oracle.com.
1 Get Started

About Adaptive Intelligent Commerce and Marketing

**Note:** The content in this guide about commerce functionality is meant only for customers using version 19.10 of the application. To find out which version you’re using, click About on the user menu.

Oracle Adaptive Intelligent Apps for Customer Experience is a managed Software as a Service (SaaS) offering. It helps you optimize your company’s revenue for commerce transactions and marketing orchestrations through adaptive, intelligent recommendations. Integrating with your commerce and marketing applications, Adaptive Intelligent Apps uses decision science and machine-learning algorithms to continuously adapt and deliver personalized product recommendations and promotions to consumers.

To understand how consumers are behaving on your commerce site in real time, the application updates, analyzes, and anonymizes data from your commerce and marketing systems at frequent intervals.

The following figure shows the flow of data to generate and continually improve the recommendations:

1. Key data sources are available to create recommendations for both signed-in and anonymous users:
   - Anonymized consumer data from your commerce and marketing applications, including real-time clickstreams
   - A wealth of third-party data from Oracle Data Cloud

2. Decision science algorithms generate personalized offers on your commerce site and in emails based on this consumer data.

3. Consumer responses to the recommendations feed back to the algorithms, creating a continuously adaptive and self-learning system.

4. Merchandisers and marketers can monitor offer and marketing performance and then:
   - Set price, inventory, and brand exclusivity policies to control which products are recommended.
   - Manually adjust the frequency of product and promotion recommendations and the frequency of routing a customer through a program path using boosts and constraints.
   - Prevent certain recommendations and paths entirely.
You can find information about the features, concepts, and how merchandisers and marketers use the application in *Using Oracle Adaptive Intelligent Commerce and Marketing*.

**About Oracle Cloud and Oracle Adaptive Intelligent Apps**

Oracle Cloud is an enterprise cloud for businesses. Oracle Cloud offers self-service business applications delivered on an integrated development and deployment platform with tools to extend and create new services rapidly. To get started, see *Get Started with Oracle Cloud Applications*.

Use the Applications Console in Oracle Cloud to:

- Activate your order
- Sign in for the first time
• Access your service instance

**Note:** You can’t manage users through the Applications Console. Manage users directly through Oracle Identity Cloud Service (IDCS). To access IDCS from Oracle Adaptive Intelligent Apps, select **User Administration** on the user menu.

---

### Version Compatibility

Oracle Adaptive Intelligent Apps for CX is supported on devices with a width of 768 pixels or higher. For detailed information on Oracle's browser support policy, see [http://www.oracle.com/technetwork/indexes/products/browser-policy-2859268.html](http://www.oracle.com/technetwork/indexes/products/browser-policy-2859268.html).

The following table lists version compatibility for the supported integrations.

<table>
<thead>
<tr>
<th>Integrated Application</th>
<th>Version Requirement</th>
</tr>
</thead>
<tbody>
<tr>
<td>Oracle Commerce Cloud</td>
<td>Oracle Commerce Cloud Release 17.2 or later</td>
</tr>
<tr>
<td>Oracle Commerce Platform</td>
<td>Oracle Commerce Platform (formerly ATG Web Commerce) Release 10.2 or later</td>
</tr>
<tr>
<td>Other integration (API-based)</td>
<td>Standard REST service calls (no version)</td>
</tr>
<tr>
<td>Oracle Commerce Experience Manager</td>
<td>Oracle Commerce Experience Manager version 3.1.2</td>
</tr>
<tr>
<td>Oracle Responsys</td>
<td>Oracle Responsys 18B or later</td>
</tr>
</tbody>
</table>

### Roles, User Administration, and Data Security

Roles and privileges control the access that users have to different features of the application. Data security controls role-based access using authentication.

#### Roles

The following table lists the role that you can assign and the privileges associated with the role.

<table>
<thead>
<tr>
<th>Role</th>
<th>Privileges</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Oracle Adaptive Intelligent Marketing User</td>
<td>• View Home page&lt;br&gt; • View programs&lt;br&gt; • Manage Connections&lt;br&gt; • Manage Supervisory Controls</td>
<td>Coordinates and supervises all activities related to Oracle Adaptive Intelligent Apps for Customer Experience, Marketing.&lt;br&gt; • Manages data source connections&lt;br&gt; • Adjusts paths to boost or constrain paths on program switches</td>
</tr>
</tbody>
</table>
The following table applies to you if you’re using version 19.10 of the application. To find out which version you’re using, click About on the user menu. The table describes the roles that you can assign and the privileges associated with them.

<table>
<thead>
<tr>
<th>Role</th>
<th>Privileges</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Oracle Adaptive Intelligent CX User</td>
<td>• View Home page</td>
<td>Coordinates and supervises all merchandising and marketing activities related to Oracle Adaptive Intelligent Apps for CX. Monitors current performance through lift analysis and adjusts recommendations using boosts and constraints for individual products, brands, categories, and promotions. Controls supervisory policies, such as price, inventory, and brand exclusivity rules. Adjusts paths to boost or constrain paths on program switches.</td>
</tr>
<tr>
<td></td>
<td>• Manage Insights</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Manage Supervisory Controls</td>
<td></td>
</tr>
<tr>
<td>Oracle Adaptive Intelligent CX Operations Manager</td>
<td>• View Home page</td>
<td>Coordinates and supervises all activities related to the operation of Oracle Adaptive Intelligent Apps for CX, such as managing data source connections, and configuring templates for commerce widgets and email campaigns. This role inherits privileges of the User role for commerce and marketing activities.</td>
</tr>
<tr>
<td></td>
<td>• Manage Connections</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Manage Data Loads</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Manage Insights</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Manage Supervisory Controls</td>
<td></td>
</tr>
</tbody>
</table>

**User Administration**

You can manage users in Oracle Identity Cloud Service (IDCS) and assign them to specific roles, which determine which tasks they can perform. If you have the appropriate privileges, select the User Administration shortcut on the user menu to go to IDCS, where you can map users to roles.

See Manage Oracle Identity Cloud Service Users for information about user administration.

**Data Security**

Oracle Identity Cloud Service uses multifactor authentication, encryption, and custom allowlists. For more information, refer to the following topics in Administering Oracle Identity Cloud Service:

- Understanding Multifactor Authentication
- Understanding Network Perimeters
- Understanding Administrator Roles

**Data Sources**

Before using Adaptive Intelligent Apps for CX, you must set up your data sources to the commerce, marketing, or sales applications appropriate for your implementation. Copy URLs and keys from the integrating applications to this page to set up the connections. After connecting, Adaptive Intelligent Apps for CX updates, analyzes, and anonymizes data from your integrating application at frequent intervals.
For more information on connecting to your integrating application, see the following topics:

- Import Data from Oracle Responsys
- Import Data from Oracle Commerce Cloud
- Import Data from Oracle Commerce Platform
- Import Data from Other Commerce Applications

For information about data sources for Adaptive Intelligent Sales, see *Implementing Adaptive Intelligent Sales*.

### Privacy Regulations and Data Protection

This topic introduces important aspects related to privacy, security, and data protection.

#### Privacy Regulations

Some jurisdictions, such as the European Union with its General Data Protection Regulation (GDPR), require special control to maintain privacy of personal information. Oracle Adaptive Intelligent Apps for CX has capabilities to help you comply with these rules. For example, you can set site-specific shopper consent requirements and consent notifications.

Refer to the following topics for more information:

- Site-Enabled Shopper Consent
- Marketing Consumer Consent
- Shopper Consent Notifications
- Explanation of Data Usage for Commerce Storefronts
- Consumers' Right to be Forgotten

#### Passwords and Private Keys

Passwords, API keys, and client secrets are masked in the user interface to prevent visibility without a deliberate action, such as clicking a link or button.

#### Display of Consumer Data

Any consumer data that's collected displays in the user interface only as anonymous data without personal information. This data is combined with audience attributes from Oracle BlueKai. Consumer data is collected only when explicit consent is either given by the shopper or not required for the storefront.

#### Site-Enabled Shopper Consent

If a storefront is governed by a jurisdiction with special requirements, an operations manager can indicate that shopper consent is required for each specified storefront. When a storefront requires shopper consent, no adaptive intelligent recommendations will be sent or shown to consumers, nor any data collected, until notification of consent is received.

The following screen capture shows the Data Sources page after a connection is established to Oracle Commerce Cloud.

(For connections to other commerce applications, use the REST API for setting the shopper consent requirement. Refer to the *REST API for Oracle Adaptive Intelligent Apps for CX* for additional information.)
Because shopper profiles aren’t associated to specific sites in your commerce application, if one or more of your sites require shopper consent, then consent rules will be applied across all sites. To help you better understand what happens when you select your Shopper Consent options, the following table illustrates the data ingestion rules for shopper profiles and orders for initial and ongoing ingestion for different scenarios.

<table>
<thead>
<tr>
<th>Scenario</th>
<th>Site A</th>
<th>Site B</th>
<th>Shopper Profile Data Ingested</th>
<th>Order Data Ingested</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Disabled</td>
<td>Disabled</td>
<td>None. No data ingested until at least one site is enabled. This is the default state for new connection.</td>
<td>None. No data ingested until at least one site is enabled. This is the default state for new connection.</td>
</tr>
<tr>
<td>2</td>
<td>Enabled</td>
<td>Disabled</td>
<td>All registered profiles. Registered profile data is ingested over all sites because shopper profiles are not associated to specific sites.</td>
<td>All orders data from Site A because it is the only enabled site.</td>
</tr>
<tr>
<td>3</td>
<td>Enabled</td>
<td>Enabled</td>
<td>All registered profiles. Registered profile data is ingested over all sites because shopper profiles are not associated to specific sites.</td>
<td>All orders data from both sites.</td>
</tr>
<tr>
<td>4</td>
<td>Enabled</td>
<td>Disabled</td>
<td>For initial ingestion, only the registered profiles that don’t contain European Union (EU) shipping or billing addresses.</td>
<td>For initial ingestion, only orders from Site A that are associated with non-EU shipping or billing addresses.</td>
</tr>
</tbody>
</table>
Marketing Consumer Consent

Adaptive intelligent models use consumer profiles and their behavioral attributes to predict the best time and medium for marketing communication. The models use machine learning to make decisions based on the type of promotional content and delivery options available. For example, the models may suggest that a consumer would prefer a marketing SMS over an email.

It's important that data ingestion takes account of consumer consent information stored in your marketing application. Let's look at two scenarios.

1. You store the consent in the system columns and the consent covers both marketing and machine learning. For example, you store consent in the EMAIL_PERMISSION_STATUS_ column in Oracle Responsys. Data ingestion automatically handles the consent and there's nothing you need to do. When consumers remove consent, their data is removed from the application during the next ingestion.

2. You store the machine learning consent and marketing consent in separate custom columns in the marketing application. For example, you might store consent in two columns: OPTED_IN = Y/N and AI_CONSENT = Y/N. In this scenario:
   - You must export a list filter from your marketing application to exclude consumers who haven't provided specific consent. For details on how to export a list filter, sign in to Oracle Responsys, navigate to Help, and search for the topic Exporting Data.
   - If a consumer first provides consent and later removes consent, use the REST endpoint to delete consumer data from marketing or adaptive intelligent models. See the topic Consumers’ Right to be Forgotten for details.
Explanation of Data Usage for Commerce Storefronts

You can place text provided by Oracle on your site in accordance with privacy regulations to explain how Oracle Adaptive Intelligent Apps collects and uses consumer information and cookies. You can copy text from the Oracle Adaptive Intelligence Registry page for this purpose.

Shopper Consent Notifications

If one or more of your sites is set to require shopper consent, then no actions will be taken by the application, and no recommendations will be delivered to a consumer, until the system receives the JavaScript call specifying that the shopper has either given consent or that it isn't required.

If you want shoppers on your sites receive notifications of their consent status, you can modify your storefront interface to implement a notification mechanism. You can use the consent mechanism you've implemented to send JavaScript calls to Oracle Adaptive Intelligent Apps using the PUT function of your dedicated secure REST endpoint. This notification informs the application of each site visitor's consent status and the time it was provided, whether consent was given, revoked, or whether consent isn't required. This status value then determines whether the application will collect any information or deliver recommendations to that consumer.

Use the JavaScript call to set the consent status for your widgets or pop-ups by calling the function that sets one of the allowable values. For example:

```javascript
consent$.AioClickStream.aio_consent('NOT_GRANTED');
```

The following table describes the allowable values for this function.

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GRANTED</td>
<td>Consumer has given consent.</td>
</tr>
<tr>
<td>NOT_GRANTED</td>
<td>Consumer has not given or revoked consent.</td>
</tr>
<tr>
<td>NOT_REQUIRED</td>
<td>Consent isn't required for the consumer.</td>
</tr>
</tbody>
</table>

Consumers' Right to be Forgotten

If you want to give your consumers the right to be forgotten, you can use the REST endpoint to request the deletion of any collected data for that consumer. When the application receives this request, it schedules a job to delete all personal data for that consumer within 24 hours of the current system time. In addition, it anonymizes any past history of clickstream events and orders. If the consumer returns to the storefront later, the rules for data collection would be the same as a new user.

You can use the REST endpoint in your code, in a form similar to this cURL examples.
For commerce:

```
curl -X DELETE --header 'Accept: application/json' 'https://<host>:<port>/offers/rest/v1/consumers/
<consumer_ID>'
```

For marketing:

```
curl -X DELETE --header 'Content-Type: text/plain' --header 'Accept: application/json' -d '<list of
consumer ids>' 'http://<host>:<port>/marketing/rest/v1/management/consumers?system=<system>'
```

For more information about this and other operations, see REST API for Oracle Adaptive Intelligent Apps for CX. For information about using a JavaScript set the consent status for your widgets or pop-ups, see Shopper Consent Notifications.
2 Import Commerce Data

About Commerce Integrations

Oracle Adaptive Intelligent Apps for CX integrates with Oracle commerce applications and non-Oracle commerce applications. Depending on your integration, a different extent of manual configuration is necessary for both data import, scheduling, and getting your widgets set up for click tracking.

**Note:** For all integration approaches, before or during the data import process, personal information about consumers is stripped out, encrypted, or anonymized to protect consumer privacy.

The following sections summarize the different supported integrations.

**Oracle Commerce Cloud**

The integration with Oracle Commerce Cloud is registered as a trusted integration and requires minimal setup. After connecting to Oracle Commerce Cloud, automated processes import and synchronize consumer, product, brand, and other data. For widget configuration, you download predefined templates and upload them to Oracle Commerce Cloud where they are ready to collect real-time clickstream data for recommendations.

*See [Import Data from Oracle Commerce Cloud](#) for more information.*

**Oracle Commerce Platform**

The integration with Oracle Commerce Platform provides plug-ins to extract consumer, product catalog, and order data into flat files. The plug-ins strip any personally identifiable information about consumers from the files, and transform the data into the adaptive intelligent data model. Because the plug-ins transform the data model, you don’t need to map fields or write your own scheduling jobs.

*See [Import Data from Oracle Commerce Platform](#) for more information.*

**Other Commerce Applications**

Oracle Adaptive Intelligent Apps for CX supports integrating with any application platform that can access its REST API to upload or retrieve data and record consumer responses. Integration requires field mapping, coding API requests to upload and synchronize data, and creating scheduling jobs. The application provides samples of widget configuration code you can use to copy into your widgets.

*See [Import Data from Other Commerce Applications](#) for more information.*

**Email Integrations**

Use the Email Templates page to design widgets for use in your email campaigns. You select the layout and styles you want, generate new images for open-time optimization, and copy the HTML into your email templates.

*See [About Email Widget Templates](#) for more information.*
Import Data from Oracle Commerce Cloud

To import data from Oracle Commerce Cloud, you must perform the following steps:

1. In the Oracle Commerce Cloud Service administration area, register Oracle Adaptive Intelligent Apps for CX as follows:
   a. On the Settings page, click **Web APIs**.
   b. On the Registered Applications page, click **Register Application**.
   c. Enter a unique name, such as **Oracle AIApps CX**.
   d. Click **Save**.

2. Locate and copy your API key as follows:
   a. In the Registered Applications list, click the name you added, and then click **Click to Reveal**.
   b. Select and copy the entire value in the Application Key field, and then click **Cancel** to close the window.
   Paste the value to a temporary location so you can access it later.

3. Obtain the value for the service endpoint URL and save it temporarily so you can access it in the next step.
   The service endpoint URL is your agent URL, such as https://www.mystore.com, as shown in the following figure.

![Add Commerce Cloud Connection](image)

4. Connect to Oracle Commerce Cloud as follows:
   a. On the Connections page in Oracle Adaptive Intelligent Apps for CX, click **Oracle Commerce Cloud**.
   b. Enter the API key and the URL value you obtained.
   c. Click **Test Connection**.
   d. Click **Continue**.

5. Select the **Enabled** check box for each site that you want enabled for adaptive intelligence.

6. For each enabled site, select whether the site must require consent from shoppers before collecting any data about them, such as their clicks and order history.
This setting determines data protection rules. For more information about shopper consent, see *Site-Enabled Shopper Consent*.

7. Click **Save**. The automated process will start to receive data from your site. As it receives data, your products, brands, categories, and promotions are reflected on the Insights page.

### Import Data from Oracle Commerce Platform

Importing data from Oracle Commerce Platform requires setup steps. You can choose one of the following approaches:

- Download and use plug-ins to extract your data into a data warehouse and upload to Oracle Storage Cloud.
- Extract your data manually and implement REST API calls for initial and ongoing uploads.

This section describes the first option, as illustrated in the following figure. For information about using the REST API directly, refer to *REST API for Oracle Adaptive Intelligent Apps for CX*.

The Oracle Commerce data warehouse plug-in extracts catalog, order, and profile data and saves it into flat files. The adaptive intelligence plug-in reads the data files, strips out personally identifiable information about consumers, and sends the output to the Oracle Storage Cloud server. A scheduled job moves the data to the adaptive intelligence database. When you start the plug-in, it looks for new or changed data files in the output subfolders that you created for the Oracle Commerce data warehouse plug-in.

![Diagram of data flow from Oracle Commerce Platform to Oracle Storage Cloud to Adaptive Intelligence and Marketing]

For more information and configuration steps, see *Summary of Configuration Steps*.

### Summary of Configuration Steps

Importing data from Oracle Commerce Platform requires the following general steps described in detail in the following topics:

1. **Configure the data warehouse plug-in**
2. **Configure the Commerce Platform server**
3. **Configure the adaptive intelligence plug-in**
Chapter 2
Import Commerce Data

Configure the Oracle Commerce Data Warehouse Plug-in

Perform the following steps in preparation for using the plug-in for extracting data:

1. If you don’t already use a data warehouse, create a database schema on a separate server to avoid placing strain on the production server.
   Refer to Creating the Data Warehouse Schema in the Oracle Commerce Business Intelligence Installation and Configuration guide, for details.
2. Download and install the Oracle Commerce Data Export Utility.
   a. Accept the license agreement on the Oracle Commerce Data Export Utility Downloads page on the Oracle Technology Network, and then click the download link.
   b. Extract the contents of the Oracle-Commerce-Data-Export-Utility.zip file to a temporary location.
   c. Extract the contents of the config.zip and src.zip files to the Oracle Commerce data warehouse server.
3. On the Oracle Commerce data warehouse server, locate the following three configuration files in the /config/atg/reporting/datawarehouse/service folder:
   o CatalogFileLogger.properties
   o OrderFileLogger.properties
   o ProfileFileLogger.properties
4. For each of these three files, change the value for defaultRoot to an absolute path and remove the carrot (^) character.
   For example, you might change the value for ProfileFileLogger.properties from defaultRoot^=/atg/dynamo/service/DWDataCollectionConfig.defaultRoot to default_root=/app/oracle/product/aiacs/atg_output/profile.
   **Note:** Ensure that you keep the default values for all other properties in the files.
5. Create the three subfolders to match each of the defaultRoot values you set in the properties files. For example:
   o /app/oracle/product/aiacs/atg_output/catalog
   o /app/oracle/product/aiacs/atg_output/order
   o /app/oracle/product/aiacs/atg_output/profile

Configure the Commerce Platform Server

To properly retrieve data from Commerce Platform, you must perform the steps outlined in the Oracle Commerce Data Export Utility attachment available on My Oracle Support (document ID 2254809.1).

Perform the following steps in addition to the steps in that document:

1. Enable SHA-256 and MD5 formats for email addresses by adding the following two lines to the JSONOutputCustomizer.properties file.
2. Increase the maxDepthMap OrderRepository value in the ProcessorPersistenceConfiguration.properties file from 4 to 5.
3. Add the following lines to the OrderResponseGenerator.properties file to exclude shipping and payment groups from Commerce.
   
   ```
   atg.commerce.order.CommerceIdentifier#paymentGroups,
   atg.commerce.order.CommerceIdentifier#shippingGroups,
   ```
4. Verify that the ExportLogRotationMessageSink.properties file defines three lines for output folders (one for each entity type). For example:
   
   ```
   defaultRootByFilePrefix=
   order_=/app/oracle/product/aiacs/ATG_TEST/atg_output/order,\n   catalog_=/app/oracle/product/aiacs/ATG_TEST/atg_output/catalog,\n   user_=/app/oracle/product/aiacs/ATG_TEST/atg_output/profile
   ```
5. Verify that the DeploymentDWDataCollectionConfig.properties file is set as enabled and specifies a valid log folder. Refer to Configuring an Asset Management Server in the Commerce Business Intelligence Installation and Configuration Guide for details.
6. Verify that the OrderSubmitLoader component is configured with scheduling properties. Refer to the Data Loader Components section in the Commerce Business Intelligence Installation and Configuration Guide for details.

Configure the Adaptive Intelligence Plug-In

The following steps must be performed by a user with the Operations Manager role.

1. Sign in to Oracle Adaptive Intelligent Apps for CX.
2. On the Connections page click Oracle Commerce Platform.
3. Enter your storefront URL and select the check box to proceed, and then click Save.

   This step associates the application with your Cloud Storage Cloud location, creates your unique credentials, and prepares the built-in scheduler to retrieve data.
### Import Commerce Data

Set the storefront URL used by the built-in scheduler to retrieve data from your Oracle Storage Cloud location. Download the plug-in and the properties file containing your unique credentials and move them to the Oracle Commerce Platform server.

#### Storefront URL


#### Oracle Storage Cloud Authentication URL

3bae24bcc1f393017b6587aa5b3123b386a376e2

#### Previous Authentication URL

72c74ad79b592043ea427cbbd49121a3b4a282d2

---

4. Click **Download Plug-In** and save the ATGIngestionClient.jar file to a temporary location.

5. Click **Download Properties**, and save the aiacs.properties file to the same location.

6. Move the JAR file to a folder on the Commerce Platform server, for example:

   /app/oracle/product/aiacs/ATG_TEST/atg_aiacs/client/target

7. Move the properties file to the same folder or a separate folder, for example:

   /app/oracle/product/aiacs/ATG_TEST/atg_aiacs/client/etc

8. Edit the properties file to change the default values for your environment as shown in the following table.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Default Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>order.properties.filename</td>
<td>OrderFileLogger.properties</td>
<td>Name of the properties file used to load consumers' product order data.</td>
</tr>
<tr>
<td>profile.properties.filename</td>
<td>ProfileFileLogger.properties</td>
<td>Name of the properties file used to load consumer profile data.</td>
</tr>
<tr>
<td>catalog.properties.filename</td>
<td>CatalogFileLogger.properties</td>
<td>Name of the properties file used to load product catalog data.</td>
</tr>
<tr>
<td>properties.file.dir</td>
<td>{enter folder}</td>
<td>Full path to the folder containing the configuration files for the data</td>
</tr>
<tr>
<td></td>
<td></td>
<td>warehouse plug-in, for example, /app/oracle/product/atg/ATG/</td>
</tr>
</tbody>
</table>
Chapter 2
Import Commerce Data

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Default Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>home/servers/atg_dw_loader_lockserver/localconfig</td>
<td>Based on the assumption that the data warehouse properties file always stay in the /atg/reporting/databwarehouse/service folder, the additional path for this property should be from the root folder to /atg. For example, if the ATG properties files are at absolute path: D:/Data/ATG/impl/config/atg/reporting/databarehouse/service, then properties.file.dir should be set to D:/Data/ATG/impl/config.</td>
</tr>
</tbody>
</table>

Note: Ensure that you don’t change the value for the cloudstorage.url property. This value is the autogenerated authentication URL containing the host name of the Oracle Storage Cloud server. It also contains the name of the container used for storing data from Oracle Commerce Platform, and an encrypted access key. To update the encrypted key, reset the URL and download the updated properties file.

Start the Adaptive Intelligence Plug-In

The adaptive intelligence plug-in runs using a Java command that takes three parameters:

- Name of the properties file
- Polling period in minutes
- Properties file folder, an optional parameter that you specify if you have moved the aiacs.properties file to a different folder

To start the plug-in and run it in the background, type a command similar to these examples:

- UNIX
  
  ```
  java -jar -Dnbo.dir=etc target/ATGIngestionClient.jar aiacs.properties 15 &
  ```

- Windows
  
  ```
  java -jar -Dnbo.dir=etc target\ATGIngestionClient.jar aiacs.properties 15
  ```

where:

- **-Dnbo.dir=etc** specifies the location of the properties file if it isn't in the same location as the JAR file.
- **15** is the polling interval in minutes for the thread to pause
Tip: For convenience, you can create a script that will generate a stop script for the process, as shown in this example.

```
java -jar target/ATGIngestionClient.jar aiacs.properties 15 & aiacsPID=!

echo $aiacsPID

echo "#!/bin/bash" > aiacs_stop.sh

echo "kill $aiacsPID" >> aiacs_stop.sh

wait
```

### Extract and Import Product Path, Price, and Inventory Data

Certain data from Oracle Commerce Platform isn't automatically synchronized as part of the scheduled updates. This data includes image URL, inventory stock levels, and product prices. However, you can use the REST API to import this information by using the product IDs you get to retrieve and update data for them. For example, you can use the POST method to update a product array with groups of attributes and use them in your call to retrieve price, inventory, or path data.

The following table lists some of the attributes you can use in the /offers/rest/v1/products POST method to update existing products.

<table>
<thead>
<tr>
<th>Data</th>
<th>REST API Attributes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Price information</td>
<td>listPrice</td>
</tr>
<tr>
<td></td>
<td>salePrice</td>
</tr>
<tr>
<td></td>
<td>currency</td>
</tr>
<tr>
<td></td>
<td>currencySymbol</td>
</tr>
<tr>
<td>Image URL</td>
<td>imageURL</td>
</tr>
<tr>
<td></td>
<td>imagePath</td>
</tr>
<tr>
<td></td>
<td>Note that imageUrl takes precedence over imagePath.ImagePath is only used if site is configured with siteBasePath.</td>
</tr>
<tr>
<td>Product URL</td>
<td>productURL</td>
</tr>
<tr>
<td></td>
<td>productPath</td>
</tr>
<tr>
<td></td>
<td>Note that productURL takes precedence over productPath. ProductPath is only used if site is configured with siteBasePath.</td>
</tr>
<tr>
<td>Inventory</td>
<td>stockLevel</td>
</tr>
</tbody>
</table>

The following example payload updates the unsynchronized attributes for products 1000315 and 10065696.

```json
{
  "products": [
    {
      "productId": "1000315",
      "imageUrl": "https://www.example.com/img/shoes.jpg",
      "productURL": "https://www.example.com/shoes/product/1000315",
      "stockLevel": 100
    },
    {
      "productId": "10065696",
      "imageUrl": "https://www.example.com/img/bags.jpg",
      "productURL": "https://www.example.com/shoes/product/10065696",
      "stockLevel": 200
    }
  ]
}
```
Oracle Adaptive Intelligent Apps for CX
Implementing Adaptive Intelligent Commerce and Marketing

Chapter 2
Import Commerce Data

```
"stockLevel": 1000,
"currency": "USD",
"currencySymbol": ":",
"listPrice": 29.99
"salePrice": 19.99
},
{
"productId": "10065696",
"imageURL": "https://www.example.com/img/boots.jpg",
"productURL": "https://www.example.com/shoes/product/10065696",
"stockLevel": 500,
"currency": "USD",
"currencySymbol": ":",
"listPrice": 63.00,
}
```

Refer to REST API for Oracle Adaptive Intelligent Apps for CX for more information:

Load Initial Data

Loading initial data imports all consumer profiles, product orders, and catalog details, such as products, promotions, and brands. The process strips personal information about consumers before uploading data by extracting only data mapped to the adaptive intelligence tables. Uploaded data is stored in the output folder you specified when configuring the data warehouse plug-in.

The first time you load data, you must trigger the initial load process in the Commerce Platform component browser by invoking the `doWalk` method to generate the data files. Subsequent updates are performed using scheduled jobs. Perform the steps outlined in the Oracle Commerce Data Export Utility attachment available on My Oracle Support (document ID 2254809.1).

**Note:** You must load catalog data before loading order data. Loading in reverse sequence will fail because the order entity type is dependent on the catalog entity type.

Schedule Data Extracts

Configure Oracle Commerce Platform to extract at a frequency that results in files that are typically no larger than 100MB. You can configure rotation using one of these two methods:

- Automatic rotation based on the number of entries written to the file (export logger property)
- Scheduled rotation, such as once per hour (component property)

The export loggers are enabled by default and set to rotate automatically after 10000 entries. To reduce the number of entries for automatic rotation, change the value for the `dataItemThreshold` property in the following properties files to a lower number, such as 1000:

- `ProfileFileLogger.properties`
- `OrderFileLogger.properties`
- `CatalogFileLogger.properties`
To schedule rotation, refer to Configuring a Schedulable Component section of the Oracle Commerce Platform Programming Guide. This guide contains information about using the scheduler and schedule properties and invoking rotation manually on any of the FileLogger components using the rotate() method.

Note: Certain data, such as price, image URLs, and inventory, isn't automatically synchronized as part of scheduled updates. Use the REST API to import this data. Refer to Extract and Import Product Price, Path, and Inventory Data for more information.

Import Data from Other Commerce Applications

Before you load data, you must understand the data structure, data types, and columns to ensure your data properly maps to the adaptive intelligence data model.

To import data from non-Oracle commerce applications, you must follow these general steps:

1. Map your catalog data model to the adaptive intelligence canonical model.
2. Create a custom routine to export the data to map into flat files.
3. Create custom code using the REST API requests to upload data.
4. Write scheduling jobs to synchronize data on an ongoing basis.

See Field Mapping for information about the canonical data model and required fields for data loading.

See REST API for Oracle Adaptive Intelligent Apps for CX for more information about using the REST API.

Import Data from Files

If you want to import data from Google Shopping or other application whose data you exported, you can use the Data Source page to select your ingestion method for each object you specify. You can select one or a combination of ingestion methods.

1. On the Data Sources page, set the ingestion method for each data object type when establishing a connection with the data source.

   Tip: If for some reason you need to change the ingestion method later, you can return the Manage Connection page and click Edit.

2. Get the URL to use to upload data to Oracle Storage Cloud using a call similar to this cUrl example:

   ```
curl -i -X PUT "https://storage_url.oraclecloud.com/v1/Storage-123/idcs_123/cx/offers/tp/google/test.zip?temp_url_sig=a1b2c3&temp_url_expires=9999999999" --upload-file gsdata-03-09-18.zip
   ```

The following table lists the object types and source types for importing data.

<table>
<thead>
<tr>
<th>Object Type</th>
<th>Supported Source Type</th>
<th>Comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>CATALOG</td>
<td>REST, GOOGLE</td>
<td>Derived from the catalog data file.</td>
</tr>
</tbody>
</table>

Note: The Google Shopping ingestion method groups products, brands, and categories into a single catalog file.
<table>
<thead>
<tr>
<th>Object Type</th>
<th>Supported Source Type</th>
<th>Comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>SITE</td>
<td>REST, TSV</td>
<td>Must load before any other data unless a site is already defined and if only one site exists in your implementation.</td>
</tr>
<tr>
<td>CONSUMER</td>
<td>REST, TSV</td>
<td>Must import before order data.</td>
</tr>
<tr>
<td>ORDER</td>
<td>REST, TSV</td>
<td>Implicitly includes order lines.</td>
</tr>
<tr>
<td>PRODUCT</td>
<td>REST</td>
<td>Individually these entities can only be ingested through the REST API, however, they are collectively ingested with the catalog.</td>
</tr>
<tr>
<td>BRAND</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CATEGORY</td>
<td></td>
<td></td>
</tr>
<tr>
<td>PROMOTION</td>
<td>REST</td>
<td>Not supported as file import.</td>
</tr>
</tbody>
</table>

Refer to Import Google Shopping Data and Tab-Delimited File Formats for more information.

Import Google Shopping Data

You can use the Data Source page to select Google Shopping as your ingestion method for each object you specify. The Google Shopping ingestion method expects that products, brands, and categories are grouped logically together in a single catalog file before it’s uploaded and ingested.

To import Google Shopping data:

1. On the Data Sources page, in the Available section, click Other Commerce Application.
2. In the Ingestion Method list, select Google Shopping for each data object you want to import.

Prepare Your Data Files

Although Google enables saving your data in several file formats, you must use a tab-delimited format (either TXT or TSV) when importing data from Google Shopping. The TXT or TSV file is typically then compressed as a GZ, ZIP, or BZ2 file.

It’s best practice to create the file using Microsoft Excel, and then save the tab-delimited, plain text file in an ASCII format. Other best practices include:

- The first line of the file is the header and must contain attribute names as provided in Tab-Delimited File Formats
- One item per line
- Don’t include trailing tabs at the end of lines
- Don’t include any tabs or line breaks in the attribute values
- For group attributes, use colon-separated sub-attributes
- For multivalue attributes, separate each value using a comma wherever applicable

Tab-Delimited File Formats

This topic includes some sample files in tab-delimited format to use for the following data:

- Site
- Consumers
- Orders
Note that brands and categories are included with products in the catalog data file. Data for promotions isn't supported at this time.

Sites Data

<table>
<thead>
<tr>
<th>siteId</th>
<th>enabled</th>
<th>name</th>
<th>storeFrontURL</th>
<th>consentRequirement</th>
<th>catalogId</th>
</tr>
</thead>
<tbody>
<tr>
<td>siteUS</td>
<td>TRUE</td>
<td>SiteUS</td>
<td><a href="http://mysite">http://mysite</a></td>
<td>REQUIRED</td>
<td>CloudCatalog</td>
</tr>
</tbody>
</table>

Consumer Data

<table>
<thead>
<tr>
<th>consumerId</th>
<th>email</th>
<th>emailAddressSHA256</th>
<th>receiveEmail</th>
<th>shippingAddressCountry</th>
<th>shippingAddressState</th>
<th>billingAddressCountry</th>
<th>billingAddressState</th>
<th>catalogId</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Order Data

<table>
<thead>
<tr>
<th>orderId</th>
<th>site</th>
<th>status</th>
<th>orderCreationDate</th>
<th>consumerId</th>
<th>promotions</th>
<th>shippingMethod</th>
<th>shippingAddressCountry</th>
<th>billingAddressName</th>
<th>billingAddressPostalCode</th>
<th>currency</th>
<th>orderAmount</th>
<th>shippingAmount</th>
<th>taxAmount</th>
<th>billingAddressState</th>
<th>billingAddressCity</th>
<th>billingAddressState</th>
<th>billingAddressCity</th>
<th>orderLineId</th>
</tr>
</thead>
<tbody>
<tr>
<td>xo10002</td>
<td>siteUS</td>
<td>NO_PENDING_ACTION</td>
<td>2011-03-03T16:00-00:00</td>
<td>se-570030</td>
<td>Ground</td>
<td>US</td>
<td>NH</td>
<td>Exeter</td>
<td>3833</td>
<td>USD</td>
<td>301.5</td>
<td>6.5</td>
<td>0</td>
<td>US</td>
<td>NH</td>
<td>Portsmouth</td>
<td>3801</td>
<td>xci1000001</td>
</tr>
</tbody>
</table>

Google Shopping Catalog Data

| id | title | description | link | price | sale | price | sale | effective date | brand | condition | image | link | gtin | product type | adwords | labels | adwords | redirect | availability | item group id | color | google product category | mpn | size | gender | age group | shipping(price) | min | qty |
|----|-------|-------------|------|-------|------|-------|------|----------------|-------|-----------|-------|------|-----|-------------|---------|--------|---------|----------|--------------|---------------|------|-----|---------|----------|---------------|------|-----|
Pause the Ingestion Queue and Delete Data Files

You can pause or resume data ingestion or delete data files that are in the queue waiting to be imported without having to disable ingestion. For example, you might delete a file in the queue because it has old or bad data that you want to correct and then upload again. You can't delete a file that's in process or already ingested. To prevent ingestion from starting on a file in the queue, you can pause ingestion by disabling it.

To pause the ingestion queue or delete files not yet loaded:

1. On the Data Sources page for your connected commerce application, click View Status > Data Ingestion Status.
2. Click the object type whose ingestion queue you want to view.
3. To disable the ingestion queue, click the Disable toggle.
4. To delete a file, click the Delete icon in the row of the file not yet processed.

Tip: Click Refresh to see any new files uploaded to Oracle Storage Cloud.

View Past Loads

If you are loading files from a third-party commerce application, you can view the history of file loads within a specified period. Viewing history can help you diagnose data load issues and give you a view into the performance of past loads. Using a specified period, you can view the start and end times and other details that can help you with troubleshooting.

1. On the Data Sources page for your connected commerce application, click View Status > Data Ingestion Status.
2. Click the object type whose file load history you want to view.
3. Click the History tab.

Tip: Click Refresh to see the latest data loads.

Field Mapping

This topic lists the required and optional fields in the canonical data model that you will need to map to the schema of your commerce application.

If you're using the data loader plug-in provided by Oracle for extracting data from Oracle Commerce Platform, this field mapping is informational only. If you've adapted your schema, or if you're extracting data from a non-Oracle commerce application, consider which data elements you require for import and map your data accordingly.
The following objects contain the data elements you’re most likely to want to import:

- Sites
- Products
- Product Categories
- Brands
- Consumers
- Orders and Order Lines
- Promotions

Refer to the following sections for the fields to use for data ingestion, their data type, and their equivalent in Oracle Commerce Cloud and Oracle Commerce Platform. Refer to the REST API for Oracle Adaptive Intelligent Apps for CX for additional information.

**Note:** The field names from Oracle Commerce Cloud and Oracle Commerce Platform are for information purposes only. They may be useful when determining the mapping to use for another commerce application data model.

## Sites

The following table describes the available fields for site data. Site data must exist prior to loading other data.

* = Required field.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Data Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>siteId</td>
<td>String</td>
<td>The unique identifier for the site.</td>
</tr>
<tr>
<td>siteName</td>
<td>String</td>
<td>The name for the site.</td>
</tr>
<tr>
<td>minStockThreshold</td>
<td>Integer</td>
<td>The minimum stock level allowed for products to be recommended.</td>
</tr>
<tr>
<td>minPriceThreshold</td>
<td>Number</td>
<td>The minimum price allowed for products to be recommended.</td>
</tr>
<tr>
<td>consentTimeThreshold</td>
<td>Integer</td>
<td>The consent time threshold to us across all sites in milliseconds.</td>
</tr>
<tr>
<td>catalogId</td>
<td>String</td>
<td>The name or list of names of the catalog of products, brands, and product categories.</td>
</tr>
<tr>
<td>enabled</td>
<td>Boolean</td>
<td>When multiple sites are configured, true if the site is enabled for adaptive intelligence.</td>
</tr>
<tr>
<td>storeFrontURL</td>
<td>String</td>
<td>The base path of the commerce server for relative URLs.</td>
</tr>
<tr>
<td>consentRequirement</td>
<td>String</td>
<td>The status of the shopper consent requirement for the site: UNKNOWN, 'REQUIRED', 'NOT REQUIRED'</td>
</tr>
</tbody>
</table>
Product Fields

The following table describes the available fields for product data.

* = Required field.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Data Type</th>
<th>Description</th>
<th>Commerce Cloud</th>
<th>Commerce Platform (ATG)</th>
<th>Google Shopping</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>productId</strong> *</td>
<td>String</td>
<td>The unique identifier for the product.</td>
<td>id</td>
<td>id</td>
<td>id</td>
</tr>
<tr>
<td><strong>name</strong> *</td>
<td>String</td>
<td>The product name.</td>
<td>displayName</td>
<td>displayName</td>
<td>title</td>
</tr>
<tr>
<td>description</td>
<td>String</td>
<td>The product description.</td>
<td>description</td>
<td>description</td>
<td>description</td>
</tr>
<tr>
<td>imageURL</td>
<td>String</td>
<td>The image URL of the product. Derived by prepending the site base path to the imagePath value.</td>
<td>N/A</td>
<td>N/A</td>
<td>imageLink</td>
</tr>
<tr>
<td>imagePath</td>
<td>String</td>
<td>The path to the product image (excluding the http://). For example, /ccstore/v1/images/?source=/file/v630864629923567673/products/myProduct.jpg. Required for email integrations.</td>
<td>primaryFullImageUrl</td>
<td>largeImage &gt; url</td>
<td>N/A</td>
</tr>
<tr>
<td>productURL</td>
<td>String</td>
<td>The URL to the page that shows the product. Required for email integrations. (Derived from productPath with prepended site base path)</td>
<td>N/A</td>
<td>N/A</td>
<td>link</td>
</tr>
<tr>
<td>productPath</td>
<td>String</td>
<td>The path to the page that shows the product (excluding the http://).</td>
<td>route</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>brandName</td>
<td>String</td>
<td>Name of the product brand.</td>
<td>brand</td>
<td>brand</td>
<td>brand</td>
</tr>
<tr>
<td>tags</td>
<td>String</td>
<td>A comma-separated list of keywords associated with the product. For example, a T-shirt might have these tags: blue,cotton,short sleeve,men's,outdoor.</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>categories</td>
<td>Array</td>
<td>A list of categories that the product belongs to.</td>
<td>parentCategories (comma-separated)</td>
<td>parentCategories (&quot;item-ref&quot; in the URL shape, not &quot;id&quot;)</td>
<td>productType</td>
</tr>
</tbody>
</table>
## Import Commerce Data

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Data Type</th>
<th>Description</th>
<th>Commerce Cloud</th>
<th>Commerce Platform (ATG)</th>
<th>Google Shopping</th>
</tr>
</thead>
<tbody>
<tr>
<td>newProduct</td>
<td>Boolean</td>
<td>If true, the payload will be filtered to include only new products.</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>inStock</td>
<td>Boolean</td>
<td>If true, the payload will be filtered to include only products currently in stock.</td>
<td>derived from inventory &gt; totalStockLevel</td>
<td>N/A</td>
<td>derived from availability</td>
</tr>
<tr>
<td>stockLevel</td>
<td>Integer</td>
<td>Number of product items currently in stock.</td>
<td>inventory &gt; totalStockLevel</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>active</td>
<td>Boolean</td>
<td>If true, then product is active.</td>
<td>active</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>activeDate</td>
<td>Integer</td>
<td>The date that the product becomes active.</td>
<td>arrivalDate</td>
<td>dateAvailable &gt; time</td>
<td>N/A</td>
</tr>
<tr>
<td>expiryDate</td>
<td>Integer</td>
<td>The date on which the product expires from the catalog. After this date, the product will no longer be recommended.</td>
<td>N/A</td>
<td>N/A</td>
<td>expirationDate</td>
</tr>
<tr>
<td>currency</td>
<td>String</td>
<td>The currency code for the product price in ISO format, for example, USD.</td>
<td>priceListGroup &gt; currencyCode</td>
<td>N/A</td>
<td>price.currency</td>
</tr>
<tr>
<td>currencySymbol</td>
<td>String</td>
<td>The currency symbol for the product price, for example, $.</td>
<td>priceListGroup &gt; symbol</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>listPrice</td>
<td>Number</td>
<td>The list price of the product. Used as the minimum price if maxListPrice is set to a value higher than the list price.</td>
<td>listPrices</td>
<td>N/A</td>
<td>price.value</td>
</tr>
<tr>
<td>maxListPrice</td>
<td>Number</td>
<td>The maximum list price of the product.</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>salePrice</td>
<td>Number</td>
<td>The sale price of the product. Used as the minimum price if maxSalePrice is set to a value higher than the sale price.</td>
<td>salePrices</td>
<td>N/A</td>
<td>salePrice.value</td>
</tr>
<tr>
<td>maxSalePrice</td>
<td>Number</td>
<td>The maximum sale price of the product.</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>salePriceStartDate</td>
<td>String</td>
<td>The start date of the product sales period in ISO-8601 format.</td>
<td>priceListGroup &gt; startDate</td>
<td>N/A</td>
<td>salePriceEffectiveDate</td>
</tr>
</tbody>
</table>
## Import Commerce Data

### Field Name | Data Type | Description | Commerce Cloud | Commerce Platform (ATG) | Google Shopping
--- | --- | --- | --- | --- | ---
soldPriceEndDate | String | The end date of the product sales period in ISO-8601 format. | priceListGroup > endDate | N/A | N/A

### Field Name | Data Type | Description | Commerce Cloud | Commerce Platform (ATG) | Google Shopping
--- | --- | --- | --- | --- | ---
soldStatus | String | The status of the product related to sold. Allowable values: ON_SALE, NOT_ON_SALE, SOME_ON_SALE | N/A | N/A | N/A

### Field Name | Data Type | Description | Commerce Cloud | Commerce Platform (ATG) | Google Shopping
--- | --- | --- | --- | --- | ---
boostOrConstrainLevel | Integer | The whole number representing the adjusted level in the range -1 to +1 that overrides default recommendations. | N/A | N/A | N/A

### Field Name | Data Type | Description | Commerce Cloud | Commerce Platform (ATG) | Google Shopping
--- | --- | --- | --- | --- | ---
neverOffer | Boolean | If true, prevent the product from being recommended. | N/A | N/A | N/A

### Category Fields

The following table describes the available fields for category data.

* = Required field.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Data Type</th>
<th>Description</th>
<th>Commerce Cloud</th>
<th>Commerce Platform (ATG)</th>
<th>Google Shopping</th>
</tr>
</thead>
<tbody>
<tr>
<td>categoryId *</td>
<td>String</td>
<td>The unique identifier of the category.</td>
<td>id</td>
<td>id</td>
<td></td>
</tr>
<tr>
<td>parentCategoryId</td>
<td>String</td>
<td>The identifier of the parent category.</td>
<td>Derived from childCategories</td>
<td>parentCatalog &gt; id (comma-separated)</td>
<td></td>
</tr>
<tr>
<td>code</td>
<td>String</td>
<td>The code representing the category.</td>
<td>N/A</td>
<td>N/A</td>
<td></td>
</tr>
<tr>
<td>name *</td>
<td>String</td>
<td>The display name of the category.</td>
<td>displayName</td>
<td>displayName</td>
<td></td>
</tr>
<tr>
<td>description</td>
<td>String</td>
<td>The category description as it appears on the storefront.</td>
<td>description</td>
<td>description</td>
<td></td>
</tr>
<tr>
<td>imageURL</td>
<td>String</td>
<td>The image URL of the category.</td>
<td>N/A</td>
<td>N/A</td>
<td></td>
</tr>
<tr>
<td>imageAltText</td>
<td>String</td>
<td>The hover text for the category image.</td>
<td>N/A</td>
<td>N/A</td>
<td></td>
</tr>
</tbody>
</table>
### Brand Fields

The following table describes the available fields for brand data.

**Note:** Brand is optional as an object. If brand information is passed with products, the brand will be created if it doesn't already exist.

*Required field.*

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Data Type</th>
<th>Description</th>
<th>Commerce Cloud</th>
<th>Commerce Platform (ATG)</th>
</tr>
</thead>
<tbody>
<tr>
<td>brandId *</td>
<td>String</td>
<td>The unique identifier for this brand.</td>
<td>brand</td>
<td>brand</td>
</tr>
<tr>
<td>Name *</td>
<td>String</td>
<td>The brand name.</td>
<td>brand</td>
<td>brand</td>
</tr>
<tr>
<td>imageURL</td>
<td>String</td>
<td>The image URL of the brand.</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>imageAltText</td>
<td>String</td>
<td>The hover text for the brand image.</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>boostOrConstrainLevel</td>
<td>Integer</td>
<td>The whole number representing the adjusted level in the range -1 to +1 that overrides default recommendations.</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>neverOffer</td>
<td>Boolean</td>
<td>If true, prevent the product from being recommended.</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>sites</td>
<td>Array</td>
<td>Comma-separated list of site identifies associated with the brand.</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>additionalAttributes</td>
<td>Array</td>
<td>List of additional attributes.</td>
<td>N/A</td>
<td>N/A</td>
</tr>
</tbody>
</table>
**Consumer Fields**

The following table describes the available fields for consumer data.

* = Required field.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Data Type</th>
<th>Description</th>
<th>Commerce Cloud</th>
<th>Commerce Platform (ATG)</th>
</tr>
</thead>
<tbody>
<tr>
<td>consumerId *</td>
<td>String</td>
<td>The unique identifier of the consumer.</td>
<td>id</td>
<td>id</td>
</tr>
<tr>
<td>shippingAddressCity</td>
<td>String</td>
<td>The city of the consumer’s shipping address.</td>
<td>shippingAddress &gt; city</td>
<td>N/A</td>
</tr>
<tr>
<td>shippingAddressCountry</td>
<td>String</td>
<td>The ISO 3166-1 country code of the consumer’s shipping address.</td>
<td>shippingAddress &gt; country</td>
<td>N/A</td>
</tr>
<tr>
<td>shippingAddressState</td>
<td>String</td>
<td>The state or region of the consumer’s shipping address.</td>
<td>shippingAddress &gt; state</td>
<td>N/A</td>
</tr>
<tr>
<td>shippingAddressPostalCode</td>
<td>String</td>
<td>The postal code of the consumer’s shipping address.</td>
<td>shippingAddress &gt; postalCode</td>
<td>N/A</td>
</tr>
<tr>
<td>receiveEmail</td>
<td>Boolean</td>
<td>Whether the consumer is willing to receive notifications, true or false.</td>
<td>receiveEmail</td>
<td>receiveEmail</td>
</tr>
<tr>
<td>emailAddressMD5</td>
<td>String</td>
<td>The consumer’s email address, MD5 hashed.</td>
<td>email</td>
<td>email</td>
</tr>
<tr>
<td>emailAddressSHA256</td>
<td>String</td>
<td>The consumer’s email address, SHA-256 hashed.</td>
<td>email</td>
<td>email</td>
</tr>
<tr>
<td>billingAddressCountry</td>
<td>String</td>
<td>The consumer’s billing address country code in ISO 3166-1 alpha-2 format.</td>
<td>paymentGroups &gt; {0} &gt; country</td>
<td>paymentGroups &gt; {0} &gt; country</td>
</tr>
<tr>
<td>billingAddressState</td>
<td>String</td>
<td>The consumer’s billing address state.</td>
<td>paymentGroups &gt; {0} &gt; state</td>
<td>paymentGroups &gt; {0} &gt; state</td>
</tr>
<tr>
<td>billingAddressCity</td>
<td>String</td>
<td>The consumer’s billing address city.</td>
<td>paymentGroups &gt; {0} &gt; city</td>
<td>paymentGroups &gt; {0} &gt; city</td>
</tr>
<tr>
<td>billingAddressPostalCode</td>
<td>String</td>
<td>The consumer’s billing address postal code.</td>
<td>paymentGroups &gt; {0} &gt; postalCode</td>
<td>paymentGroups &gt; {0} &gt; postalCode</td>
</tr>
<tr>
<td>shopperConsent</td>
<td>String</td>
<td>Value indicating the shopper’s consent status. Allowable values are GRANTED, NOT_GRANTED, and NOT_REQUIRED.</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>shopperConsentTimestamp</td>
<td>Integer</td>
<td>Time the consumer set the shopper consent status in milliseconds.</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>additionalAttributes</td>
<td>Array</td>
<td>List of additional attributes.</td>
<td>N/A</td>
<td>N/A</td>
</tr>
</tbody>
</table>
Order Fields

The following table describes the available fields for order data. All fields are optional.

Some important notes about orders and order lines:

- Don’t include order lines with negative product quantities (these are used by merchants as refunds)
- Where possible, provide order data in chronological order.
- Break large files into smaller files while still ensuring complete orders with order lines are together within one file.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Data Type</th>
<th>Description</th>
<th>Commerce Cloud</th>
<th>Commerce Platform (ATG)</th>
</tr>
</thead>
<tbody>
<tr>
<td>orderId</td>
<td>String</td>
<td>The unique identifier of the order.</td>
<td>id</td>
<td>id</td>
</tr>
<tr>
<td>orderCreationDate</td>
<td>Integer</td>
<td>The date the order was created in ISO 8601 format. Provide orders in chronological order.</td>
<td>creationTime</td>
<td>creationTime</td>
</tr>
<tr>
<td>consumerId</td>
<td>String</td>
<td>The unique identifier of the consumer (registered or guest profile ID).</td>
<td>profileId</td>
<td>profileId</td>
</tr>
<tr>
<td>orderLineCount</td>
<td>Integer</td>
<td>The number of order lines.</td>
<td>commerceItems</td>
<td>commerceItems</td>
</tr>
<tr>
<td>promotions</td>
<td>String</td>
<td>The comma-delimited list of promotion IDs used on the order.</td>
<td>From /ccagent/v1/orders/ + orderId + ? includeResult=full endpoint adjustments &gt; pricingModel &gt; id</td>
<td></td>
</tr>
<tr>
<td>status</td>
<td>String</td>
<td>Free-form status of the order. INCOMPLETE is a reserved status.</td>
<td>state</td>
<td>stateAsString</td>
</tr>
<tr>
<td>shippingMethod</td>
<td>String</td>
<td>The shipping method of the order.</td>
<td>shippingGroups &gt; {0} &gt; shippingMethod</td>
<td></td>
</tr>
<tr>
<td>currency</td>
<td>String</td>
<td>The currency code in ISO 4217 format of the amounts of the order. Inherited by all order lines.</td>
<td>priceInfo &gt; currencyCode</td>
<td></td>
</tr>
<tr>
<td>orderAmount</td>
<td>Number (Double)</td>
<td>The total order amount.</td>
<td>priceInfo &gt; total</td>
<td></td>
</tr>
<tr>
<td>shippingAmount</td>
<td>Number (Double)</td>
<td>The shipping amount.</td>
<td>priceInfo &gt; shipping</td>
<td></td>
</tr>
<tr>
<td>taxAmount</td>
<td>Number (Double)</td>
<td>The tax amount.</td>
<td>priceInfo &gt; tax</td>
<td></td>
</tr>
<tr>
<td>shippingAddressCountry</td>
<td>String</td>
<td>The shipping address country code of the order in ISO 3166–1 alpha-2 format.</td>
<td>shippingGroups &gt; {0} &gt; shippingAddress &gt; country</td>
<td></td>
</tr>
<tr>
<td>shippingAddressState</td>
<td>String</td>
<td>The shipping address state of the order.</td>
<td>shippingGroups &gt; {0} &gt; shippingAddress &gt; state</td>
<td></td>
</tr>
<tr>
<td>shippingAddressCity</td>
<td>String</td>
<td>The shipping address city of the order.</td>
<td>shippingGroups &gt; {0} &gt; shippingAddress &gt; city</td>
<td></td>
</tr>
</tbody>
</table>
## Import Commerce Data

### Field Name | Data Type | Description | Commerce Cloud | Commerce Platform (ATG)
--- | --- | --- | --- | ---
shippingAddressPostalCode | String | The postal code of the order. | shippingGroups > {0} > shippingAddress > postalCode | shippingGroups > {0} > shippingAddress > postalCode
billingAddressCountry | String | The billing address country code of the order in ISO 3166–1 alpha-2 format. | paymentGroups > {0} > country | paymentGroups > {0} > country
billingAddressState | String | The billing address state of the order. | paymentGroups > {0} > state | paymentGroups > {0} > state
billingAddressCity | String | The billing address city of the order. | paymentGroups > {0} > city | paymentGroups > {0} > city
billingAddressPostalCode | String | The postal code of the order. | paymentGroups > {0} > postalCode | paymentGroups > {0} > postalCode
site | String | The name identifier of the site. Leave blank if only one site is enabled. | N/A | N/A
orderLines | Array | The order lines of the order. | Self-constructed comma-separated list from commerceItems | Self-constructed comma-separated list from commerceItems

### OrderLine Fields

The following table describes the available fields for order line data. All fields are optional.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Data Type</th>
<th>Description</th>
<th>Commerce Cloud</th>
<th>Commerce Platform (ATG)</th>
</tr>
</thead>
</table>
| orderLineId | String | The unique identifier of the order line. Order line IDs can be scoped within an order. Two different orders can have the same order line IDs. For example, order A could have order lines 1,2,3 and order B could have order lines 1,2,3,4. | commerceItems > id | commerceItems > id
| orderId | String | The unique identifier of the parent order of the order line. | id | id
| productId | String | The unique identifier of the product of the order line. | commerceItems > productId | commerceItems > productId
### Field Name | Data Type | Description | Commerce Cloud | Commerce Platform (ATG)
--- | --- | --- | --- | ---
`productQuantity` | Integer (Double) | The product quantity. Note that this is rounded to the nearest whole integer. Negative product quantities that indicate refunds are ignored. | `commerceltems > quantity` | `commerceltems > quantity`
`currency` | String | The currency code of the amounts of the order. | `commerceltems > priceInfo > currencyCode` | `commerceltems > priceInfo > currencyCode`
`productListPrice` | Number (Double) | The product unit list price. | `commerceltems > priceInfo > listPrice` | `commerceltems > priceInfo > listPrice`
`productSalePrice` | Number | The product unit sale price if the product is on sale. | `commerceltems > priceInfo > salePrice` | `commerceltems > priceInfo > salePrice`
`productOnSale` | Boolean | If true, the product is on sale. | `commerceltems > priceInfo > onSale` | `commerceltems > priceInfo > onSale`
`shippingAddressCountry` | String | The shipping address country code of the order line. | `shippingGroups > {0} > shippingAddress > country` | N/A
`shippingAddressState` | String | The shipping address state of the order line. | `shippingGroups > {0} > shippingAddress > state` | N/A
`shippingAddressCity` | String | The shipping address city of the order. | `shippingGroups > {0} > shippingAddress > city` | N/A
`billingAddressCountry` | String | The billing address country code of the order. | N/A | N/A
`billingAddressState` | String | The billing address state of the order. | N/A | N/A
`billingAddressCity` | String | The billing address city of the order. | N/A | N/A

### Promotion Fields
The following table describes the available fields for promotion data. All fields are optional.

### Field Name | Data Type | Description | Commerce Cloud | Commerce Platform (ATG)
--- | --- | --- | --- | ---
`promotionId` | String | The unique identifier for the promotion. | `id` | `id`
`typeCode` | String | The promotion type code. | `type` | `type`
`type` | String | The promotion type. | `templateValues > discount_type_value` | `templateValues > discount_type_value`
`includedProducts` | String | A comma-separated list of product identifiers in the promotion. | Derived from XML contents pmdlRule attribute | `PSC_value > includedProducts`
`productCount` | Integer | The number of products in the promotion. | N/A | N/A
### Field Name | Data Type | Description | Commerce Cloud | Commerce Platform (ATG)
--- | --- | --- | --- | ---
name | String | The promotion name. | N/A | N/A
description | String | The promotion description. The value for this field is the text displayed in the widget. | description | description
enabled | Boolean | If true, the promotion is enabled. | enabled | enabled
startDate | Integer | The date the promotion starts. | startDate | startDate
derendDate | Integer | The date the promotion ends. | endDate | endDate
imageURL | String | The image URL of the promotion. | N/A | N/A
boostOrConstrainLevel | Integer | The whole number representing the adjusted level in the range -1 to +1 that overrides default recommendations. | N/A | N/A
neverOffer | Boolean | If true, prevent the product from being recommended. | N/A | N/A
sites | Array | Comma-separated list of site identifiers associated with the promotion. | N/A | N/A
additionalAttributes | Array | List of additional attributes. | N/A | N/A

## View Data Ingestion Status

Viewing data ingestion status enables you to monitor the status of data being pulled into the application directly from your commerce system or manually from data files. You can view the number of objects ingested by object type as shown in the following screen capture.

To view data ingestion status:

1. On the Data Sources page for your connected commerce application, click **View Status > Data Ingestion Status**.
**Tip**: Refresh the browser to see the latest data loads. Most data is polled every fifteen minutes. Inventory and stock levels from Oracle Commerce Cloud are checked more frequently, typically every five minutes.

2. If you are loading files from a third-party commerce application, you can:
   - Click the link in the Start Time column for the object type you want to view the latest load details for that object type.
   - Click an object type to view the files currently in the queue and the history of file loads within a specified period.

If you're importing data from Oracle Commerce Cloud, you will see only the basic information.

The following table describes the status columns.

<table>
<thead>
<tr>
<th>Column</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data Object</td>
<td>The object types appropriate for the connected commerce application. These may vary depending source of the data. For example, for Google Shopping data, products, brands, and categories are combined into a single Catalog object type. Another example is for ingestion of tab-delimited (TSV) data files, where the table lists site data, a prerequisite to other data. If you ingest data using the data file loader, for example for TSV files, you can view more details about historical data loads.</td>
</tr>
<tr>
<td>Last Loaded</td>
<td>The date and time of the last successful data load, even if no data is new or changed.</td>
</tr>
<tr>
<td>Total Records</td>
<td>The total number of records in the corresponding data load.</td>
</tr>
<tr>
<td>Skipped Records</td>
<td>The number of records excluded from the data load, typically in compliance of any special regulations preventing the ingestion of personal information.</td>
</tr>
<tr>
<td>Errors</td>
<td>For data loads whose errors are five percent or more of the total records, the number of errors. Errors under five percent are not reported and don't prevent a successful data load.</td>
</tr>
<tr>
<td>Status</td>
<td>Indicates either completion of the data load (errors under five percent) or failure (errors exceeding five percent).</td>
</tr>
</tbody>
</table>
View Clickstream Status

Viewing clickstream status enables you to monitor the status of events within the last seven days for a specified site. You can view data for various events, such as the number of times users viewed products or logged on to a commerce site.

To view clickstream data:

1. On the Data Sources page for your connected commerce application, click **View Status > Clickstream Status**.
2. Select the site for the clickstream data you want to view.

3. To view the latest clickstream data, click **Refresh**.
**Tip:** If no clickstream events were ingested for an event type within the last seven days, the count shows as zero, even if they were ingested previously. The zero count makes it easy to view which event types have no recent ingestion activity.

The following table describes the clickstream event types.

<table>
<thead>
<tr>
<th>Event Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Any Page Load</td>
<td>Number of times consumers arrive on any page with a page ID.</td>
</tr>
<tr>
<td>BlueKai Profile Match</td>
<td>Number of distinct consumers who accessed the site and have an existing BlueKai ID associated with their consumer ID.</td>
</tr>
<tr>
<td>Category Page Load</td>
<td>Number of times consumers loaded a category page on the site.</td>
</tr>
<tr>
<td>Content Clicks</td>
<td>Number of times consumers clicked a link inside any content area on the site</td>
</tr>
<tr>
<td>Home Page Load</td>
<td>Number of times consumers loaded the Home page on the site.</td>
</tr>
<tr>
<td>Item Added to Cart</td>
<td>Number of times consumers added items to their carts.</td>
</tr>
<tr>
<td>Item Removed from Cart</td>
<td>Number of times consumers removed items from their carts.</td>
</tr>
<tr>
<td>Order Confirmed</td>
<td>Number of times a product order was confirmed and purchased on the site.</td>
</tr>
<tr>
<td>Product Page Load</td>
<td>Number of times consumers loaded the Home page on the site.</td>
</tr>
<tr>
<td>Search Results Page Load</td>
<td>Number of times consumers loaded the Search Results page on the site.</td>
</tr>
<tr>
<td>Site Entry</td>
<td>Number of times any consumers entered the site.</td>
</tr>
<tr>
<td>User Login</td>
<td>Number of times consumers logged in to the site.</td>
</tr>
<tr>
<td>User Logout</td>
<td>Number of times consumers logged out of the site.</td>
</tr>
</tbody>
</table>
3 Import Marketing Data

About Marketing Integrations

Integrate Oracle Responsys and Oracle Adaptive Intelligent Apps for Customer Experience to get real-time bi-directional data transfer. When you add adaptive intelligent switches to your marketing programs, the switches send and receive decisions. You also activate scheduled data imports from Oracle Responsys. These imports provide an ongoing feed of marketing behavior to train the machine-learning models, giving you continuous improvements to performance.

If you have multiple Oracle Responsys accounts, you can connect all of them to a single Oracle Adaptive Intelligent Apps for CX tenant. You can then send prediction requests and publish programs for all the accounts.

See also: Import Data from Oracle Responsys.

Import Data from Oracle Responsys

You must complete steps both in Oracle Responsys and Oracle Adaptive Intelligent Apps to connect to and import data from Oracle Responsys. Use this flow chart for a summary of steps that you must do in both applications.
To import data from Oracle Responsys:

1. Sign in to Oracle Responsys and find the connection information:
   a. Click Account.
   b. On the Account Management page, click Responsys Adaptive Intelligence Connectivity in the Adaptive Intelligence section.
   c. On the Responsys – Adaptive Intelligence Connectivity page, copy values from the following fields in the Responsys Connection Information section:
      - Account Name
      - Service Endpoint URL
      - API Key: If the API key isn't visible, click Edit, and then click Generate New Key to view the key.
      - Host Name and User Name under Responsys File Area

2. Sign in to Oracle Adaptive Intelligent Apps for Customer Experience (CX) to paste the information that you copied:
   a. Click Connections and then click Data Sources.
   b. Click Oracle Responsys.
   c. In the Add Responsys Connection dialog box, paste the information you copied from Oracle Responsys.
d. Enter the Oracle Responsys file area path where you want to export the files for Adaptive Intelligent Apps. For example, /download/aiapps. This action creates a new folder in the specified path when you successfully connect to Oracle Responsys and upload the SSH2 key.

If you’re importing data from more than one Oracle Responsys account, you can select a different file area path for each of your accounts.

e. Click Connect.

The connection is saved. You see the Data Sources page showing the connection information for Oracle Adaptive Intelligence Apps. If you don’t see the connection information, click the Manage Connection button for the Oracle Responsys connection to view it.

3. Copy the following information from the Oracle Adaptive Intelligence connection information:
   - Oracle Adaptive Intelligence API Key
   - Service Endpoint URL
   - Security Endpoint URL

4. Click Download SSH2 Key to download the public key file to upload it on Oracle Responsys. A .pub file containing the key is downloaded.

   Note: If you want to import data from multiple Oracle Responsys accounts, use the same SSH2 key.

5. On the Oracle Responsys Account Management page, register Oracle Adaptive Intelligent Apps for CX:
   a. Click Responsys – Adaptive Intelligence Connectivity in the Adaptive Intelligence section.
   b. On the Responsys – Adaptive Intelligence Connectivity page, paste values that you copied from the Oracle Adaptive Intelligence Connection Information section.
   c. Click Test Connection and confirm that the connection is successful.

6. Upload the SSH2 key file to Oracle Responsys:
   a. On the Account Management page, click Manage SSH2 Keys in the Account Customization section.
      If you don’t have the option to manage SSH2 keys, send your downloaded SSH2 key to Oracle Support so that they can upload it.
   b. On the Manage SSH2 Keys page, add the key:
      i. Select the SCP user from the list.
      ii. Add a description of the key in the Tag name for key field.
      iii. Click Select from the Select Key field and upload the SSH2 key from the download location.
      iv. Click Add key.

      The key appears in the list of keys on the page. The status shows processing and changes to completed after an approval process in Oracle Responsys. Oracle Adaptive Intelligent Apps creates the folder based on the Responsys file area path that you entered on the Add Responsys Connection dialog box.

7. Export contacts list, app channel list, and feed data from Oracle Responsys. For details, see Export Data from Oracle Responsys.

If you want to import data from another Oracle Responsys account, follow the same steps and use the SSH2 key that you already downloaded.
Export Data from Oracle Responsys

After setting up your connection to Oracle Responsys, you must configure regular exports of the contact list, app channel list, and feed data. Adaptive intelligence models make better predictions when they have more data available. Exporting data from Oracle Responsys regularly ensures that the adaptive intelligence models have the latest data.

To prepare the exports:

1. Create an export job for the contacts list.
2. Create an export job for the app channel list.
3. Configure a feed data export.

Create an Export Job for the Contacts and the App Channel Lists

**Note:** You need to export the app channel list only if you have push notifications.

1. Sign in to Oracle Responsys.
2. Click **Data**, and then click **Connect**.
3. On the Manage Connect page, click **Create Job**.
4. In the Create Job dialog box, select **Export Data** as the job you want to create.
5. Enter a name for the export job.
   - For contacts list, enter **AI Apps Contacts List Export**.
   - For app channel list, enter **AI Apps App Channel List Export**.

This screenshot shows an example of exporting the contacts list.
6. Optionally, provide a description.
7. Click **Create**.

The export job is created.

You don’t want the export job that you’re creating to expire. Click the edit icon for **Expiration** and select **Never expires** in the Edit expiration date dialog box.

This screenshot shows the **Expiration** edit icon for contacts list export.
8. In the Select Object section:
   a. Select List if you're exporting contacts list. Select App channel list if you're exporting app channel list.
   b. Select the Only export records inserted or updated since the time of last job run check box.
9. Click Destination Connectivity.
10. In the Destination Connectivity section:
    a. Select Responsys File Server from the list.
    b. Change the path to the path that you entered while creating the connection.
11. Click Destination Specification.
12. In the Destination Specification section, complete the fields as indicated in this table.

<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>File to be created (prefix, name of the file, suffix and file extension)</td>
<td>○ No prefix&lt;br&gt; ○ CONTACT_LIST_ or APPCHANNEL_LIST_ (depending on which list you're exporting)&lt;br&gt; Don't enter any other name for the file.&lt;br&gt; ○ YYYYMMDD_HH24MISS</td>
</tr>
</tbody>
</table>
13. Click **Notification**.
14. Optionally, add a notification to the job so that you know if the job fails.
15. Click **Schedule**.

You can repeat this process for all contact lists and app channel lists that you want to export. Ensure that you set the exports for slightly different times of the day.

This screenshot shows an example of the schedule for contacts list export.

<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Field</td>
<td>o .txt</td>
</tr>
<tr>
<td>Character set</td>
<td>Unicode (UTF-8)</td>
</tr>
<tr>
<td>Field delimiter</td>
<td>Tab</td>
</tr>
<tr>
<td></td>
<td>Tab is the recommended delimiter. You can also select your preferred delimiter.</td>
</tr>
<tr>
<td>Field enclosure</td>
<td>“</td>
</tr>
<tr>
<td>Insert column header as first line</td>
<td>Select</td>
</tr>
<tr>
<td>Encryption/compression</td>
<td>Compress to file (zip or gz)</td>
</tr>
<tr>
<td></td>
<td><strong>Compress and encrypt file with PGP/GPG Key</strong> is not supported. If you want to encrypt your exports, contact Oracle Support.</td>
</tr>
<tr>
<td>Additional ready file at completion of download</td>
<td>Do not create</td>
</tr>
</tbody>
</table>
16. In the Schedule section:
   a. Select **Recurring** and **Daily**.
   b. Select a date and time that works for your account activity, and set the recurring frequency to every day with no end date.
   c. Click **Activate** to save and activate the export job.

Configure a Feed Data Export

Let’s look at two scenarios:

- Add a file location to any existing export jobs for contact events. If the contact events are partially exported in one job, and partially exported in another, follow the steps described in the section Add a File Location to an Existing Export Job.
- Set up a new feed export for any contact events that you don’t currently export. See the section Set Up a New Export Feed for details. If the contact events are partially exported in one or many jobs, and the rest are not exported at all, see the section Add a File Location to an Existing Export Job. Follow these steps on all existing export jobs.

Add a File Location to an Existing Export Job

1. Navigate to the Responsys Connect data export tool.
2. On the Connect page, filter the job type to **Export Feed Data** and click **Submit**.
3. Select your existing export job and ensure that it’s exporting events for **Sent**, **Open**, **Click**, **Convert**, and **Launch State** at the minimum.
4. Navigate to **Destination Connectivity**.
5. Click **Add location**.
6. With the new file location highlighted, select **Responsys File Server** from the list.
7. Change the path to the path that you entered while creating the connection.
8. Click the arrow and complete all the steps in the rest of the process.
   - Set the Destination Specification section according to the specifications detailed in the section Create an Export Job for the Contacts List. Set the schedule to six times a day. If you’re using different settings for your current export, contact Oracle Support.
9. Click **Save and Close**.

**Set Up a New Export Feed**

1. Navigate to the Responsys Connect data export tool and select **Export Feed Data Job** to create an export job.
2. Select the event files **Sent, Open, Click, Convert, and Launch State**.
3. Click **Destination Connectivity**.
4. In the Destination Connectivity section, select **Responsys File Server** from the list.
5. Change the path to the path that you entered while creating the connection.
6. Click **Destination Specification**.
7. Use this table to complete the Destination Specification section.

<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Character set</td>
<td>Unicode (UTF-8)</td>
</tr>
<tr>
<td>Field delimiter</td>
<td>Tab</td>
</tr>
<tr>
<td></td>
<td>Tab is the recommended delimiter. You can also select your preferred delimiter.</td>
</tr>
<tr>
<td>Field enclosure</td>
<td>“</td>
</tr>
<tr>
<td>Insert column header as first line</td>
<td>Select</td>
</tr>
<tr>
<td>Encryption/compression</td>
<td>Compress to file</td>
</tr>
</tbody>
</table>
### Import Marketing Data

<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Compress and encrypt file with PGP/GPG Key</td>
<td>is not supported. If you want to encrypt your exports, contact Oracle Support.</td>
</tr>
<tr>
<td>Additional ready file at completion of download</td>
<td>Do not create</td>
</tr>
</tbody>
</table>

8. Click **Notification**, and optionally add a notification to the job.
9. Click **Schedule**. If your schedule isn't already set, set it to six times a day.
10. Click **Activate** to save and activate the export job.

See also:
- *Exporting Contact Event Data*
- *Understanding Responsys Connect*
4 Configure Oracle BlueKai

Connect Audiences for Oracle BlueKai

If you use Oracle BlueKai, you can activate individual products to send data to BlueKai about consumers who responded to product offers. You can use the categories of consumers for the products that receive responses, such as clicks, add-to-carts, and purchases, in your marketing campaigns or to create look-alike audiences for retargeting. After you configure rules for adaptive intelligence, your campaigns will include consumer data for the activated products.

File a Service Request

Before you can enable this feature, you must file a service request with Oracle BlueKai to enable your connection. After your request is fulfilled, you must configure your connection as described in the next section.

Connect to Oracle BlueKai

Connecting to Oracle BlueKai enables activation of individual products, which queues data for sending to BlueKai.

1. On the Data Sources page, click Oracle BlueKai.
2. Enter your BlueKai partner ID. This value is the ID associated with your partner seat.
3. Enter your web service user key. This value is the developer key (bkuid) from the BlueKai web service key tool.
4. Enter the web service private key. This value is the authentication key (bksecretkey) from the BlueKai web service key tool.
5. Click Test Connection.
6. Click Connect.

Activate Products of Interest

Now you are ready to activate individual products. Activating a product for the first time creates classifications in the BlueKai taxonomy for the connections you have set up. Your taxonomy will contain a new classification named Adaptive-Intelligence-Classification shown in the following screen shot:
The classification contains commerce product recommendations, with subclassifications for adds-to-cart, clickers, and purchasers for the activated product.

**Configure BlueKai Campaigns**

To use this data for connected audiences, configure your rules and campaigns in BlueKai using the adaptive intelligence classifications for those products.
5 Configure Web Widgets

Web Widget Templates

Control how your customers receive recommendations on your commerce storefronts using web widget templates. If you integrate with an email service provider, you can also use the email widget code to include personalized recommendations in your email campaigns.

Most widgets use machine learning to control recommendations displayed on your commerce site. Other widgets, such as recently viewed and trending items, use basic recommendation algorithms that don’t require advanced machine learning.

These predefined widget templates give you a starting place for your designs.

<table>
<thead>
<tr>
<th>Widget Type</th>
<th>Purpose</th>
<th>Uses Adaptive Intelligence</th>
</tr>
</thead>
<tbody>
<tr>
<td>Product (Single)</td>
<td>Shows individual product recommendations</td>
<td>Yes</td>
</tr>
<tr>
<td>Product (Carousel)</td>
<td>Shows groups of product recommendations</td>
<td>Yes</td>
</tr>
<tr>
<td>Promotion</td>
<td>Shows individual promotions</td>
<td>Yes</td>
</tr>
<tr>
<td>Also Bought (Carousel)</td>
<td>Shows a group of items that other shoppers bought together with the highlighted product</td>
<td>No</td>
</tr>
<tr>
<td>Also Viewed (Carousel)</td>
<td>Shows a group of items that other shoppers viewed together with the highlighted product</td>
<td>No</td>
</tr>
<tr>
<td>Recently Viewed (Carousel)</td>
<td>Shows a group of items that the shopper viewed within a recent time period</td>
<td>No</td>
</tr>
<tr>
<td>Trending Items (Carousel)</td>
<td>Shows a group of items that have current purchasing trends</td>
<td>No</td>
</tr>
<tr>
<td>Clickstream</td>
<td>Collects clickstream activity to feed into the adaptive intelligence models. Not applicable for widget design or displaying recommendations.</td>
<td>Not applicable</td>
</tr>
</tbody>
</table>

**Note:** For all widgets, you specify details, such as the display title, either directly in the widget code or in the commerce application. Any specifications you make for a widget type apply to all widgets of that type.

Recently Viewed Carousel

You can use the Recently Viewed widget to remind shoppers about products that they previously viewed. This gives you another chance to influence the shopper’s decision to purchase. When a shopper purchases a product that’s also in the Recently Viewed carousel, that product is then removed from the carousel.
Clickstream tracking in the widget collects information about what products shoppers view and when they view them. The tracking is slightly different for registered and anonymous shoppers.

- For registered shoppers, tracking is in the current and previous sessions for the same browser across all devices.
- For anonymous shoppers, tracking is the same as for registered shoppers, except only on the same device.

By default, the Recently Viewed widget tracks up to thirty items viewed within the last three months. To change these values, you must use the REST API. See Set Tracking Period and Count for Recently Viewed Items in REST API for Oracle Adaptive Intelligent Apps for CX.

### Trending Items Carousel

You can use the Trending Items widget to highlight products that are gaining popularity. The recommendation algorithms identify trending items by comparing products purchased in a specified interval to the previous interval. Suppose your widget uses the weekly trending interval. The carousel shows the products with the highest percentage increase in purchases between the last seven days and the previous seven days. For example, on March 22, it shows the products with the highest growth in sales from 9-15 March to 16-22 March.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>interval</td>
<td>(Required) The trending period to calculate trends</td>
<td>• DAY (24 hours)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• WEEK (7 days)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• MONTH (30 days)</td>
</tr>
<tr>
<td>n</td>
<td>(Required) The number of trending items to display</td>
<td>Any whole integer</td>
</tr>
<tr>
<td>minOrderThreshold</td>
<td>(Optional) The minimum number of times a product must be ordered in the previous time interval for it to be considered in trending calculations. You can use this value to filter out certain scenarios. For example, if there’s an increase from 1 item sold to 2 items sold, it could appear as a top-trending item.</td>
<td>Any whole integer. Default value is 0.</td>
</tr>
<tr>
<td>priceThreshold</td>
<td>(Optional) The minimum price so that only products above this price threshold are considered for trending. If you have a product price policy set on the Policies page, it will filter out the products under that price. So only products whose prices are greater than both the policy and the widget’s price threshold are considered.</td>
<td>A number value in the format for your default currency. For example, 19.99.</td>
</tr>
</tbody>
</table>

If you’re using Oracle Commerce Cloud, you can set these parameters in the widget design. Otherwise, you can use the REST API calls in your widget code. This cURL example uses these specifications to derive trending items:

- Compare the weekly intervals
- Display 10 items
- Consider only 5 or more product orders for trending
Configure Web Widgets

• Consider only products over 12.99


Configure Widgets for Oracle Commerce Cloud

For each widget, you create an extension ID from Oracle Commerce Cloud, enter it on the Commerce Widgets page to generate templates. Download the templates and then upload them to Oracle Commerce Cloud. You can choose the following widget templates:

• Clickstream
• Product (Single)
• Product (Carousel)
• Promotion
• Recently Viewed (Carousel)
• Trending Items (Carousel)
• Also Bought Items (Carousel)
• Also Viewed Items (Carousel)

If you have multiple commerce storefronts or sites, you have the option of enabling one or all sites for adaptive intelligence. You can use any of your widgets on all the sites that you enable. Before configuring widgets, ensure that filtering is enabled as described in the next section.

Configuration Steps

Each widget you configure must have an extension ID. Perform the following steps for each widget to configure:

1. In the Oracle Commerce Cloud Service administration area, create an extension ID as follows:
   a. On the Settings page, click Extensions.
   b. Click the Developer tab.
   c. Click Generate ID.
   d. In the New Extension ID window, enter a unique name for the widget, and then click Save.
   e. Copy the newly generated extension ID.
      You will paste this value in the next step.

2. In Oracle Adaptive Intelligent Apps for CX, associate the extension ID with a widget type and get the widget code as follows:
   a. On the navigation menu, under Connections, select Commerce Widgets.
Configure Web Widgets

b. Ensure that the value in the Platform list is Oracle Commerce Cloud.

c. Enter the newly generated extension ID for the widget type you want to configure.

d. Click Download Widget.

e. Save the downloaded zip file to your file system.

3. In Oracle Commerce Cloud, upload the zip file as follows:

a. On the Settings page, click Extensions.

b. Click the Installed tab.

c. Click Upload Extension.

d. Locate the zip file, and then click Open.
4. Make any required modifications to fit your specific usage or in-house style or specific using the editor in Oracle Commerce Cloud.

Refer to the Modifying Uploaded Extension Widgets section in Using Oracle Commerce Cloud Service for more information.

5. For promotion widgets, see Configure Promotions for Oracle Commerce Cloud for steps to enable them for display and click tracking.

Adaptive Intelligence Settings

For additional settings, such as setting the number of products to display in the carousel, display text, timeout period, or filtering, perform the following steps for each widget:

1. In Oracle Commerce Cloud, click Design in the navigation menu.
2. For each widget that you want to configure, select the widget and then click the Widget Settings icon.
3. To change the default display text for the widget, enter the text you want in the Title field.
4. To change the number of items to show for carousel widgets, change the number in the Number of Products field. For new widgets, the default number of items is 12.
5. To change the timeout period, change the value in the Timeout Period field. The default timeout period is 6000 milliseconds (6 seconds).
6. To use restrictions to limit which recommendations are placed on a product detail page or on a collection's landing page on the storefront:
   a. Select one of these general restrictions, or select None. A selection is required.
      - Parent Collection (only products in the collection being viewed or searched)
      - Parent Brand (only products of the brand being viewed or searched)
      - On Sale
      - New - Past Month
      - New - Past Week
      - New - Past Day
   b. To restrict the shown products to a specific brand or collection, enter the ID or comma-separated brand IDs for the respective field.
7. To use exclusions to prevent recommendations on a product detail page or on a collection's landing page on the storefront:
   a. Select one of these general exclusions, or select None. A selection is required.
      - Parent Collection (exclude products in the collection being viewed or searched)
      - Parent Brand (exclude products in the collection being viewed or searched)
      - On Sale
      - New - Past Month
      - New - Past Week
      - New - Past Day
   b. To exclude one or more specific products, enter the product ID or comma-separated product IDs.
   c. To exclude products in one or more specific brands or collections, enter the ID or comma-separated IDs into the respective field.
Note:

- The adaptive intelligence models work best when no restrictions are applied so that there's sufficient opportunity to learn and react to preferences and behavior. Because restrictions and filters you use will affect recommendations, it's best practice to use them only in special circumstances.
- Depending on how you set up your recommendation widgets, be careful that you don't specify conditions that cancel each other out or affect recommendations in a way you didn't expect.
- Filtering occurs after recommendations are generated through machine-learning. Therefore, if you have any exclusions that include items already boosted, they will be hidden in recommendations using the widget where exclusions were applied.

Configure Promotions for Oracle Commerce Cloud

To configure promotions to display on your storefront and to track consumer responses through click tracking, your promotions must be associated with one or more products. You associate them using image URLs and link URLs as described in this topic.

Perform the following steps in Oracle Commerce Cloud for each promotion:

1. If you already have an image to use for the promotion, on the Media page, copy the value in the Path field. You will use this value when creating the image URL for the promotion.
2. If don't already have media for the promotion, upload media as follows:
   a. On the Media page, select either Products, Collections, or General.
   b. Click Upload and browse to select the image to upload.
   c. In the Location section, copy the value in the Path field.
      You will use this value when creating the image URL for the promotion.
   d. Click Save.
3. Click Marketing.
4. If you already have a promotion that you want to enable for display and click tracking, skip to the next step. Otherwise, create a promotion as follows:
   a. In the New Promotion list, select the promotion type you want to create.
   b. Enter a display name.
   c. Click OK
   d. Enter values for promotion fields as appropriate, and then click Create.
5. Click the promotion to configure it for click tracking.
6. In the Description section, click the Image icon.
7. In the URL field, enter the full path to your storefront URL. Append the URL with /file/ and then the path value you copied from the media page. For example, https://mystorefront.com/file/products/10DollarsOffJacket.png.
8. Optionally, add alternative text or change the size or formatting in the appropriate fields.
9. On the Link tab, enter the URL to which you want to direct consumers when they click on the widget. This is typically the product or collection detail page. For example, https://mystorefront.com/womenjackets/product/1024.
10. Optionally, select a target value, such New Window.
11. Click OK.
12. In the Availability section, ensure that there is a valid start date and select Enabled.
These values are required for the promotion widget to be available.

13. Click **Save**.

## Configure Widgets for Oracle Commerce Platform

Before you begin working with widget configuration, you must register the API used by the plug-in, modify XML files, and restart the server. The prerequisite steps set up the JavaScript and widget templates that you can copy as needed to configure your widgets. The widget configuration steps show you how to use the widget code to configure individual widgets.

### Prerequisite Steps

For initial configuration or after server updates, perform the following steps:

1. **Register the REST API `getProduct` method used by the plug-in as follows:**
   a. Locate the `ActorChainRestRegistry.properties` file and open it for editing.
      
      This file is typically in the following location:
      
      `/CommerceAccelerator/Plugins/Account/src/mail/config/atg/rest/registry/`
   
   b. Add a comma followed by a backslash character (`,\`) to the last line in the file.
   
   c. Add the following line to the end of the file:
      
      `/atg/commerce/catalog/ProductCatalogActor/getProduct`

   **Note:** Ensure that the path is the same as the path created when installing the plug-in.

2. **Add the profile ID in the profile object as follows:**
   a. Locate the `beanFilteringConfiguration.xml` file and open it for editing.
      
      This file is typically in the following location:
      
      `/app/oracle/product/atg/ATG/CommerceAccellerator/Plugins/Account/src/main/config/atg/dynamo/service/filter/bean/`
   
   b. Add the line `<property name="id"/>` into both the short and summary filters of the user item as shown here:
      
      ```xml
      <item-descriptor name="user" default-filter="short">
        <filter id="short">
          <property name="id"/>
          <property name="email"/>
          <property name="securityStatus"/>
          <property name="middleName" xml-combine="remove"/>
        </filter>
        <filter id="summary">
          <property name="id"/>
          <property name="gender"/>
        </filter>
      </item-descriptor>
      ```
   
   c. Save and close the file.

3. **Alter the promotion description to be a rich text area as follows:**
   a. Locate the `pricingModels.xml` file and open it for editing.
Configure Web Widgets

This file is typically in the following location: app/oracle/product/atg/ATG/CommerceAccelerator/Applications/B2CStore/src/main/config/atg/commerce/pricing/

b. For the descriptionDefault property at around line 28, change the data-type value from string to big string as shown here:

```xml
<property name="descriptionDefault" data-type="big string" column-name="description" category-resource="categoryBasics" display-name-resource="descriptionDefault"/>
```

c. Restart the Commerce Platform server.

4. In Oracle Adaptive Intelligent Apps for CX, copy the clickstream JavaScript as follows:

   a. On the navigation menu, under Connections, select Commerce Widgets.

   b. In the Platform list, select Oracle Commerce Platform.

   c. In the row for the Clickstream widget type, click Copy Code.

5. Paste the code into a file so you can access it easily later.

   You will paste this code on the pages to track when a consumer performs an action on your site, as described in the next section.

Widget Configuration Steps

Perform the following steps for each widget to configure:

1. In Oracle Adaptive Intelligent Apps for CX, get the widget code as follows:

   a. On the navigation menu, under Connections, select Commerce Widgets.

   b. In the Platform list, select Oracle Commerce Platform.

   c. Click Copy Code for the widget type you want to configure.

   Tip: To ensure that you don’t overwrite content on your clipboard, paste the code to a temporary text file.

2. Add the widget code to the page as appropriate for your storefront. For example, you might insert the widget code as follows:

   a. Log in to the Oracle Commerce Platform Business Control Center as an administrator.

   b. Click Workbench and then click Experience Manager.

   c. Select Web > Home Pages > Default Homepage.

   d. On the Editor tab, select mainContent, and then click Add.

   e. In the Select Cartridge window, select RichTextMain, and then click OK.

   f. In the Section Settings section, enter a unique name.

   g. In the Contents section, click Enter Text.

   h. In the Edit Text Area window, paste the copied widget code, and then click OK.

   i. Click Save.

   See Adding cartridges to rules in the Oracle Commerce Workbench User’s Guide for more information.

3. To track when a consumer performs an action on your site (such as adding or removing an item from their cart, or making a purchase) paste the clickstream code on all pages right before the closing </body> tag.

4. For promotion widgets, ensure that you have associated a name and image as required for display in the application. Refer to Field Mapping and REST API for Oracle Adaptive Intelligent Apps for CX for more information.
Configure Widgets for Other Commerce Applications

To configure widgets for non-Oracle commerce applications, you must follow these general steps summarized here:

1. Get the widget template code
2. Update the widget code with your REST service calls.
   - Add a call to your own REST service.
   - Paste the widget template code into your code.
3. Add clickstream event tracking to your site pages.

To get the widget code:

1. On the navigation menu, under Connections, select **Commerce Widgets**.

   ![Commerce Widgets](image)

   - In the Platform list, select **Other Commerce Application**.
   - Click **Copy Code** for the widget type you want to configure.

   2. In the Platform list, select **Other Commerce Application**.
   3. Click **Copy Code** for the widget type you want to configure.
Tip: To ensure that you don’t overwrite content on your clipboard, paste the code to a temporary text file.

4. Repeat these steps for each widget type, as needed.

See Create REST Service Calls in Widget Templates, Add Clickstream Event Tracking to Site Pages, and Override Default Titles for Recommendation Widgets for additional tasks to perform.

Create REST Service Calls in Widget Templates

The scripts in the widget templates contain a method named getData. If you’re using a non-Oracle commerce system, you must use the getData method to retrieve an array of recommendation IDs and pass an array of details about that product or promotion.

To use the getData method for a product or promotion:

1. Add a call to your own REST service and paste the widget template code into your code.
2. When you obtain your data, you can change the temporary values between the single quote characters with the actual values for the product or promotion.

   ```javascript
   getData: function(data) {
       var results = [], _self = this, i = 0;
       $.each(data.items, function(key, value){
         result = {
           product : {
             prodID: value,
             listPrice: '*** INSERT LIST PRICE HERE ***',
             displayName: '*** INSERT DISPLAY NAME HERE ***',
             smallImageURLs: '*** INSERT IMAGE URL HERE ***',
             route: '*** LINK TO PRODUCT HERE *** ' + '?r=' + data.id
           }
         };
         results[key] = result.product;
         if(data.items.length == ++i) _self._render(results);
       });
       this.resize();
   },
   
   For example, you might modify this in your code substituting values as shown here.

   ```javascript
   getData: function(data) {
       var results = [], _self = this, i = 0;
       $.each(data.items, function(key, value){
         $.getJSON('https://www.mysite.com/rest/service/for/your/products?
productID=value', function (myData) {
           result = {
             product : {
               prodID: value,
               listPrice: myData.price,
               displayName: myData.name,
               smallImageURLs: myData.url,
               route: myData.link + '?'r=' + data.id
             }
           };
           results[key] = result.product;
           if(data.items.length == ++i) _self._render(results);
       });
   });
   ```
To add parameters to your recommendations, such as the maximum number of items in the carousel or a custom title, or to add filters, add them directly before the `get` command. For example:

```
$.AioClickStream.aioEventListeners('3rd-party', false, false,
  function(aioRecommendData) {
    aioRecommendData.numberOfItems = 12;
    aioRecommendData.filters = [
      { usage: "filter", strategy: "productCategory", id: "PC1" },
      { usage: "exclusion", strategy: "brand", id: "Brand1,Brand2" }
    ];
    aioRecommendData.get('product', function (data) {
      ......
    });
  });
```

**Note:** The adaptive intelligence models work best when no filters are applied so that there's sufficient opportunity to learn and react to preferences and behavior. Because filters can affect recommendations, it's best practice to use them only in special circumstances.

See [Add Clickstream Event Tracking to Site Pages](#) for additional steps to perform.

## Add Clickstream Event Tracking to Site Pages

When a consumer performs an action on your site, this triggers an event to use for clickstream tracking. These events could be signing in, adding or removing an item from their cart, or making a purchase. If you're using a non-Oracle commerce system, you must add event tracking for your widgets manually as described in this topic.

1. For each page of your commerce site, add clickstream template code after jQuery is imported but before any code that sends events.

For example:

```
$(document).ready(function() {
  /*
  * Set global variables if they haven't already been set somewhere else
  * send data for site entry if it hasn't been sent somewhere else
  */
  if(!window.hasOwnProperty('$aio_settings')) {
    window.$aio_settings = {
      serverUrl: 'https://<myserver>:<port>',
      timeout: 6000
    };
    // initialize
    $.getScript("https://<host>:<port>/offers_public/js/aio_clicks.js", function() {
      aioEventListeners('3rd-party-global', false, false, function(aioClickData) {
        aioClickData.send();
      });
    });
  }
});
```
Configure Web Widgets

2. Add any other widget in place on the page where you want the widget. Insert the JavaScript for events you want tracked after the clickstream widget.

Refer to Clickstream Events for a list of events you can use along with code examples.

Override Default Titles for Recommendation Widgets

The default text that displays for recommendations is *We think you’d like* ${displayName} where ${displayName} is replaced with the product name. You can override this default text by modifying the HTML in the widget template code as described in this topic.

**Note:** If you’re using Oracle Commerce Cloud, use the predefined templates to update the Title setting for the widget you want to modify. See Configure Widgets for Oracle Commerce Cloud.

To override the default title text:

1. In the HTML you copied for your widgets, locate the tag for the title as shown in this example:

   ```html
   <div id="aio_product" class="loading">
     <div class="aio_product_item">
       <h2>We think you'd like ${displayName}</h2>
       <a href="${route}"
         <img class="aio_product_image" delay_src="${largeImageUrl}" alt="${displayName}" />
       </a>
       <div class="aio_brand_name">${brand}</div>
       <div class="aio_list_price"><span class="glyphicon glyphicon-tag" aria-hidden="true"></span> $${lowestListPrice}</div>
     </div>
   </div>
   ```

2. Substitute "We think you’d like" with the text that you want for the widget.

Clickstream Events

If you use Oracle Commerce Platform or a third-party commerce application for your widgets, you can use a variety of predefined events in your code. This table lists these events, their required attributes, and code examples.

<table>
<thead>
<tr>
<th>Event Name</th>
<th>Triggered When</th>
<th>Required Attributes</th>
<th>Code Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>LoginSuccess</td>
<td>A consumer successfully signs in</td>
<td>userid should contain the consumer's ID for the storefront</td>
<td>$.aioCommerceEvent('LoginSuccess', {userid: 'u123'});</td>
</tr>
<tr>
<td>LogoutSuccess</td>
<td>A consumer successfully signs out</td>
<td>(None)</td>
<td>$.aioCommerceEvent('LogoutSuccess');</td>
</tr>
<tr>
<td>AddedItem</td>
<td>A consumer successfully adds an item to the cart</td>
<td>cart containing a comma-separated list of all product IDs currently in the cart</td>
<td>$.aioCommerceEvent('AddedItem', {cart: '123,124,125', product: '123'});</td>
</tr>
<tr>
<td>Event Name</td>
<td>Triggered When</td>
<td>Required Attributes</td>
<td>Code Example</td>
</tr>
<tr>
<td>--------------------</td>
<td>--------------------------------------------------------------------------------</td>
<td>-------------------------------------------------------------------------------------</td>
<td>------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>RemovedItem</td>
<td>A consumer successfully removes an item from the cart</td>
<td>cart containing a comma-separated list of all product IDs currently in the cart</td>
<td>$.aioCommerceEvent('RemovedItem', {cart: '123,124,125', product: '123'});</td>
</tr>
<tr>
<td>SearchResults</td>
<td>A consumer searches the storefront</td>
<td>searchTerm containing the user's search term</td>
<td>$.aioCommerceEvent('SearchResults', {searchTerm: 'shoes'});</td>
</tr>
<tr>
<td>OrderComplete</td>
<td>A consumer completes a purchase</td>
<td>orderId containing the order ID of the completed order, cart containing a comma-separated list of all product IDs currently in the cart</td>
<td>$.aioCommerceEvent('OrderComplete', {orderId: 'o123', cart: '123,124,125'});</td>
</tr>
<tr>
<td>HomePage</td>
<td>A consumer arrives on the home page of the storefront</td>
<td>(None)</td>
<td>$.aioCommerceEvent('HomePage');</td>
</tr>
<tr>
<td>ProductPage</td>
<td>A consumer arrives on the product page</td>
<td>productId containing the product ID of the product page</td>
<td>$.aioCommerceEvent('ProductPage', {productId: 'p123'});</td>
</tr>
<tr>
<td>CategoryPage</td>
<td>A consumer arrives on the category page</td>
<td>categoryId containing the category ID of the category page</td>
<td>$.aioCommerceEvent('CategoryPage', {categoryId: 'c123'});</td>
</tr>
<tr>
<td>PageLoad</td>
<td>A consumer arrives on any page with a page ID. This is in addition to the product, category, and home page events.</td>
<td>pageload containing the unique ID of the page</td>
<td>$.aioCommerceEvent('PageLoad', {pageload: 'home'});</td>
</tr>
<tr>
<td>ContentClick</td>
<td>A consumer clicks a link inside any content area on the site</td>
<td>contentId containing the unique reference ID for the content area of the page</td>
<td>$.aioCommerceEvent('ContentClick', {contentId: 'content-area-1'});</td>
</tr>
</tbody>
</table>
About Email Templates

The provided email widget templates enable you to display widgets for product recommendations in your emails to consumers. Using the widgets, your emails will contain references to specific URLs to render the correct product image and direct consumers to the correct product page when they click recommendations.

The following figure illustrates the flow from launching email sent to consumers to the consumer clicking the link that leads to the recommended product in the storefront. Actions by consumers, such as opening email and clicking image URLs, are recorded and fed back as data for continuous machine learning.
Important aspects related to the email flow are:

- You create images for email widgets during configuration on the Connections page.
- Your email service provider contributes to click-tracking fed back to adaptive intelligence using tracking parameters appended to the URL. Recorded events include when a consumer opens email, clicks a link in email for a recommended product, adds the product to their cart, and purchases the recommended product.

Configure Email Widgets

**Note:** Your email service provider must be able to insert the consumer’s email address as a SHA256-hashed value into the image and link URLs as a text replacement field. Contact your email service provider directly to understand if they meet this minimum requirement.

To configure email widgets for integrated email applications, you must perform the following steps:

1. On the navigation menu, under Connections, select **Email Widgets**.

2. Select the number of products you want displayed.
3. Select the layout of product images that you want.
   The layout selections change depending on the number of products you selected.
4. Select the strategy you want or select **None (Default)** to use the default personalized recommendations.
   **Note:** The Also Bought and Trending strategies use simple algorithms with no personalization. If you want to use personalization, use the default instead of selecting Also Bought or Trending here. You can then modify the code to apply a product filter to show only items bought together or trending items. There’s more information about this later in this procedure.

5. Optionally, to edit styles such as font size, colors, and alignment:
   a. Click **Edit Template Styles**.
   b. On the Image tab, Name tab, and Sale Price tab, enter the same settings as your existing style, or change the style if you want.
   c. On the List Price tab, set the style you want for the list price for when items aren’t on sale. For example, you might want the list price to display as large or bold text.
   d. On the same tab, set the style you want for the list price for when items are on sale. For example, you might want the list price to display as small strike-through text.
e. Click **Save**.
f. Click **Publish Now** to immediately start generating updated images based on your changes. Or, you can wait for the scheduled update process to generate updated images for the products on your marketing application. By default, the scheduled updates are every fifteen minutes.

6. Click **Copy to Clipboard** to copy the HTML code and paste it into your email service provider’s template.

7. Replace the variables in the HTML code with appropriate values. For example, replace `$$EMAIL_HASH_SHA256_` with the appropriate code from your email service provider to insert the hashed email address value into the URLs. Optionally, add product filters as URL parameters.

8. To add other URL parameters such as product filters, see **Email Template Code** for more information.

9. If you are using Oracle Responsys, see **Configure Oracle Responsys** for information about event tracking, dynamic variables for product default values, and other considerations.

If you want to make further changes to style, edit and change the layout of the generated HTML code to make it fit with your existing email templates. Consult your email HTML designer who will be able to modify the placement of the image and link URLs provided.

### Email Template Code

The email templates provide code that defines two URL, one for the link and one to the source image file for that link. Both URLs follow the same structure with the URL path being everything before the first question mark (?) and the URL parameters, after the first question mark. The following sections describe the URL path and URL parameters to help you modify the URLs to meet your requirements.

#### Example Code

The following examples show the code provided in the template for different strategy selections.

**Default Personalization (No Strategy)**

```html
<a href="https://my-apps-domain.com/offers/widget/email/${SITE_ID}/link/1?n=${TOTAL_NUM_RECS}
&e=${EMAIL_SHA256_HASH_}&pf=${PRODUCT_FILTER}&eid=${RIID_}&d=${DEFAULT_PRODUCT_ID_1}"">
<img src="https://my-apps-domain.com/offers/widget/email/${SITE_ID}/image/1?n=${TOTAL_NUM_RECS}
&e=${EMAIL_SHA256_HASH_}&pf=${PRODUCT_FILTER}&eid=${RIID_}&d=${DEFAULT_PRODUCT_ID_1}"
</a>
```

**Also Bought Strategy**

```html
<a href="https://my-apps-domain.com/offers/widget/email/${SITE_ID}/ab/link/1?n=${TOTAL_NUM_RECS}
&e=${EMAIL_SHA256_HASH_}&pf=${PRODUCT_FILTER}&eid=${RIID_}&d=${DEFAULT_PRODUCT_ID_1}"
><img src="https://my-apps-domain.com/offers/widget/email/${SITE_ID}/ab/image/1?n=${TOTAL_NUM_RECS}
&e=${EMAIL_SHA256_HASH_}&pf=${PRODUCT_FILTER}&eid=${RIID_}&d=${DEFAULT_PRODUCT_ID_1}"
</a>
```

**Trending Strategy**

```html
<a href="https://my-apps-domain.com/offers/widget/email/${SITE_ID}/tr/link/1?n=${TOTAL_NUM_RECS}
&e=${EMAIL_SHA256_HASH_}&pf=${PRODUCT_FILTER}&eid=${RIID_}&d=${DEFAULT_PRODUCT_ID_1}"
><img src="https://my-apps-domain.com/offers/widget/email/${SITE_ID}/tr/image/1?n=${TOTAL_NUM_RECS}
&e=${EMAIL_SHA256_HASH_}&pf=${PRODUCT_FILTER}&eid=${RIID_}&d=${DEFAULT_PRODUCT_ID_1}"
</a>
```
URL Path

The URL path includes three important details:

- The unique domain used for your instance of Oracle Adaptive Intelligent Apps for CX.
  
  This value never changes and is in all your email template URLs.
  
- The type of URL, either link or image.
  
- The number representing the placement identifier, such as 1.

  This value ensures that the first recommended product appears in position one, the second in position two, and so on. For example, if you’re showing three recommended products and all URLs had a placement value of 1, then all three positions would show the exact same product.

URL Parameters

URL parameters are based on elements you can include in your URLs. The application uses these parameters to identify the consumer. If the consumer is unknown, the application uses a cached default ID to identify the consumer until it obtains enough data to create the consumer profile. The cached ID ensures the relevant content is viewed in the future until the cache is cleared.

The following table describes each URL parameter with sample values. Values for all parameters are required.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Value</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>e</td>
<td>The SHA256 hash email address of the email recipient</td>
<td>${EMAIL_SHA256_HASH_}</td>
<td>fa07e99e86f7f7a27ae3</td>
</tr>
<tr>
<td>eid</td>
<td>A unique string ID representing the email recipient and the email campaign.</td>
<td>${RIID_}${campaign.id?html}${launch.id?html}</td>
<td></td>
</tr>
<tr>
<td>d</td>
<td>The default product ID to display when the consumer email address is unknown. Increment the numeric value to represent the position in your email. Must be defined last.</td>
<td>${DEFAULT_PRODUCT_ID_#}</td>
<td>product123</td>
</tr>
<tr>
<td>n</td>
<td>The maximum number of product recommendations in the email.</td>
<td>${TOTAL_NUM_RECS}</td>
<td>6</td>
</tr>
<tr>
<td>pf</td>
<td>Product filter applied after running adaptive intelligence algorithms.</td>
<td>${PRODUCT_FILTER}</td>
<td>{&quot;strategy&quot;: &quot;alsoBought&quot;, &quot;id&quot;: &quot;product-123&quot;}</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>{&quot;strategy&quot;: &quot;trending&quot;, &quot;minPrice&quot;: 0.99, &quot;minOrders&quot;: 4, &quot;interval&quot;: &quot;WEEK&quot;}</td>
</tr>
<tr>
<td>pid</td>
<td>The ID of the product that Also Bought is based on.</td>
<td>${PRODUCT_ID}</td>
<td>product345</td>
</tr>
</tbody>
</table>
Configure Oracle Responsys

This topic describes best practices and steps in Oracle Responsys to configure event tracking, set dynamic variables for product default values, and address any responsive design considerations.

Event Tracking

It's best practice to configure event tracking in Oracle Responsys by creating a link table for your email campaign. This way you can manually add rows for each product placement position. Adding the rows directly to the link table as shown in these steps instead of generating them automatically avoids duplication of work and manual edits to HTML files.

1. Create a link table and add a row for each product placement position. Use link names that you can easily identify in your reports as shown in this example.

   ![Link tracking](image)

   1. **AI_RECS_PLACEMENT_1**
      - **Link URL**: http://your-aiapps-domain.com/offers/widget/email/link/?e={$EM...
   2. **AI_RECS_PLACEMENT_2**
      - **Link URL**: http://your-aiapps-domain.com/offers/widget/email/link/?e={$EM...
   3. **AI_RECS_PLACEMENT_3**
      - **Link URL**: http://your-aiapps-domain.com/offers/widget/email/link/?e={$EM...
   4. **textlink_View_in_browser**
      - **Link URL**: ${form(campaign.name)}

2. Within your email design in the rules editor, apply the href value for your links, using the following format:

   ```
   ${clickthrough('AI_RECS_PLACEMENT_1','SITE_ID','TOTAL_NUM_RECS','EMAIL_SHA256_HASH_','PRODUCT_FILTER','RIID_','DEFAULT_PRODUCT_ID_1','launch.id','campaign.id')}
   ```
3. Apply the values for image source URLs as you would for any other image in the email.

4. To apply styling and scaling to make the images fit within your existing template, add the appropriate HTML code. For example, `<img style="border:none;" width="320" src="img_url">

Default Values and Dynamic Variables

It's best practice to configure default values and dynamic variables on the Data Sources page in Oracle Responsys. The dynamic variables determine which recommendations to display based on the values you provide.
Note: The template code expects variables named `DEFAULT_PRODUCT_ID_#` where the `#` is replaced by the placement number. When supplying multiple values, such as IDs in `PRODUCT_FILTER`, ensure to use comma separators with no leading or trailing space characters.

Responsive Design

You can apply any mobile optimization or responsive design in Oracle Responsys, as required. For example, you might want alternative content flows for mobile devices, such as wraps, hidden, or visible. Work with your HTML designer to manage mobile optimization or responsive design within your email template.
7 Troubleshooting

Why are promotions missing on the Insights page?

Promotions must be configured to use the description information for name, image URL, and other details. Contact an operations manager or administrator to ensure promotions are configured properly on your commerce application.

Why isn't my price or inventory data being updated?

If you use Oracle Commerce Platform, your price, image URL, and inventory data isn't automatically synchronized as part of the scheduled updates. Use the REST API to import this data from Oracle Commerce Platform. Refer to Extract and Import Product Price, Path, and Inventory Data for more information.

Why aren't order or catalog changes reflected on the data warehouse server?

If the Commerce data warehouse publishing server was set up without using the automation tool, some necessary configuration might be missing. The following table lists some possible symptoms and solutions.

<table>
<thead>
<tr>
<th>Symptom</th>
<th>Solution</th>
</tr>
</thead>
<tbody>
<tr>
<td>Submitted orders are missing from the Order Submit Loader queue.</td>
<td>Verify that the ARF.base property is defined in the MODULES list in the atg_pub.config file. This file is typically found in /app/oracle/product/atg/bin/config. This configuration is required to ensure published changes by deployment generate event logs for the catalog loader queue on the data warehouse server.</td>
</tr>
<tr>
<td>Published changes by deployment aren't generating event logs for the Catalog Loader queue.</td>
<td>Verify that the following properties files exist and are set as enabled:</td>
</tr>
<tr>
<td></td>
<td>• ProductCatalogDeploymentListener.properties</td>
</tr>
<tr>
<td></td>
<td>• ProductCatalogDeploymentLogEntryGenerator.properties</td>
</tr>
<tr>
<td>Published changes by deployment aren't reflected by the data warehouse exported data.</td>
<td>Edit the production target and create an additional data warehouse agent with the RMI port and host name.</td>
</tr>
</tbody>
</table>