Sitecore CMS & DMS 6.6 or later

Engagement Analytics Configuration Reference Guide

A conceptual overview for developers and administrators

Table of Contents

Chapter 1

Introduction

This document provides configuration and usage instructions for the Sitecore Customer Engagement Platform (CEP) and Engagement Analytics. Sitecore administrators and developers should read this document before implementing Engagement Analytics.

This document provides an overview of the Engagement Analytics architecture, details of Engagement Analytics configuration, pointers to common Engagement.Analytics Application Programming Interfaces, and instructions for troubleshooting.

This information in this manual is valid for Sitecore CMS 6.6 or later and for Sitecore DMS 6.6 or later.

This document contains the following chapters:

- **Chapter 1 — Introduction**
  This introduction to the document.
- **Chapter 2 — Sitecore Engagement Analytics Overview**
  This chapter describes concepts and features in Sitecore DMS, and then explains the architecture of the Engagement Analytics engine.
- **Chapter 3 — Engagement Analytics Configuration**
  This chapter describes how to configure the Engagement Analytics engine, including databases, settings, and providers.
- **Chapter 4 — The Sitecore.Analytics API**
  This chapter describes the Sitecore.Analytics API.
- **Chapter 5 — Engagement Analytics Troubleshooting**
  This chapter contains instructions for troubleshooting issues with the Engagement Analytics engine.

Chapter 2

Sitecore Engagement Analytics Overview

This chapter describes Engagement Analytics concepts and features, and then explains the architecture of the Engagement Analytics engine.

This chapter contains the following sections:

- Engagement Analytics Features and Concepts
- Multivariate Tests
- Engagement Analytics Architecture

2.1 Engagement Analytics Features and Concepts

This section describes Engagement Analytics concepts, including instructions to implement the corresponding features.

Important
In this document, most references to the configuration file refer to entries in `/App_config/include/Sitecore.Analytics.config` on your system.

2.1.1 Page Request Tracking

The Analytics Engagement engine automatically tracks all page requests. You can cancel tracking of specific requests. Most Engagement Analytics operations involve a sequence of requests visited by a web client. With Sitecore, each page request corresponds to an item in the database. Therefore each Engagement Analytics operation is associated with a Sitecore item, also known as a page.

2.1.2 Engagement Analytics Visits

An Engagement Analytics visit represents a sequence of HTTP requests from a single web client in a single browsing session, including any events associated with those pages.

To identify a sequence of HTTP requests from a single user, Sitecore sends an Engagement Analytics session cookie to the web client. The name of the session cookie is `SC_ANALYTICS_SESSION_COOKIE`. The value of the session cookie before the first comma character (“,”) matches a value in the `VisitID` column of the `Visits` table and is used to identify the record of an engagement analytics visit in the *Analytics* database. The session cookie expires when the user closes the web client.

Engagement Analytics visits represent browser sessions. Visits often, but do not always, correspond to ASP.NET sessions. A visit can span multiple ASP.NET sessions. For example, ASP.NET terminates ASP.NET sessions after the number of minutes specified by the `timeout` attribute of the `/configuration/system.web/sessionState` element in the `web.config` file. The first request from a web client creates an Engagement Analytics visit that corresponds to the ASP.NET session. If the ASP.NET session expires before the next request, ASP.NET creates a new session, but Sitecore does not create a new visit. The single visit represents the activity of the user within and between ASP.NET sessions.

Identifying a Visitor

An Engagement Analytics visitor represents multiple individual visits from a single web client.

To identify repeat visits from a single user, Sitecore sends a persistent session cookie to the web client. The name of the persistent session cookie is `SC_ANALYTICS_GLOBAL_COOKIE`. The persistent session cookie expires one year after the last page request from the web client.

2.1.3 Visitor Tags

You can use visitor tags to store Engagement Analytics attributes about visitors, each of which represents a different site visitor. For more information about visitors, see the section *Identifying a Visitor*.

Important
Visitor tags differ from security profiles and Engagement Analytics profiles. Visitor tags apply to visitors in individual web clients rather than to users or individual ASP.NET sessions. For more information about Engagement Analytics profiles, see the section Engagement Analytics Profiles. For more information about user profiles, see the *Security Reference.*

Note
Visitor tag values apply to visitors in individual web clients. If a user accesses the web site using two different user agents, the Engagement Analytics engine records two visits with separate tag values.

2.1.4 Engagement Analytics Profiles

Engagement Analytics profiles contain information gathered from each visit. You can assign profile values to content items, automatically updating the user's profile when the user accesses that content. For more information about visits, see the section Engagement Analytics Visits.

Important
Engagement Analytics profiles differ from security profiles and visitor tags. Engagement Analytics profiles apply to individual visits in individual web clients rather than to users or visitors. For more information about visitor tags, see the section *Visitor Tags*.

Note
Engagement Analytics profile values apply to individual visits in individual web clients. If a user accesses the web site using two different user agents, the Engagement Analytics engine records two visits with separate profile values.

2.1.5 Engagement Analytics Page Events

The Engagement Analytics engine associates zero or more events with each page request. By default, each page request raises an event. You can configure user actions to raise events, such as when a user clears a form or submits their vote in a poll that uses AJAX instead of ASP.NET postback technology. System operations can generate events, such as when a timer expires or a video reaches its conclusion.

Goals and Failures

Some events can represent specific objectives of the web site owner, such as if a user accesses a resource or completes a form. Other events may represent failures, such as if a user does not click any of the links in a list of search results. Register specific types of goals and failures using the `System/Analytics/Goal` data template as described in the section *How to Register an Engagement Analytics Page Event*.

Campaigns

You can activate the campaign using the value of the `sc_camp` query string parameter or you can assign a campaign to a content item, such as a landing page. For more information about campaigns, see the section *Campaign Categories.* To change the name of the sc_camp query string parameter, see the section *The Analytics.CampaignQueryStringKey Setting*.

How to Register an Engagement Analytics Page Event

There are many predefined page events located in the `/Sitecore/System/Settings/Analytics/Page Events` item. These include common events such as Session Begin, Session End, Search, and Logout.

To register additional page events:

1. In the **Content Editor**, select the `/Sitecore/System/Settings/Analytics/Page Events` item.
2. In the **Content Editor**, insert an event definition item using the `System/Analytics/Page Event` data template.

Note
Code and configuration files reference event definition items by name.

Tip
Use patterns to identify events within components, such as `Video Starts` and `Video Concludes`.

3. In the **Content Editor**, in the event definition item, in the **Data** section, enter event parameters.

For more information about how to activate an event using a URL query string parameter, see the section *How to Trigger an Event with a URL Query String Parameter*.

Important
You must register an event before using it in any other way.

How to Trigger an Event with a URL Query String Parameter

You can set the `sc_trk` URL query string parameter to the name of an event, goal, or failure. For information about configuring the name of this parameter, see the section *The Analytics.EventQueryStringKey Setting*.

How to Map Query String Parameters to Events

To configure an arbitrary query string parameter to trigger an event:

1. Add a `<trigger>` element within the `/configuration/sitecore/pipelines/startTracking/processor` element in `web.config` that has a value of `Sitecore.Analytics.Pipelines.StartTracking.ProcessQueryString,Sitecore.Analytics` for the `type` attribute.
2. In the `<trigger>` element, set the `querystring` attribute to the name of the query string parameter.
3. In the `<trigger>` element, set the `eventname` attribute to the name of the event to trigger.

If the URL contains the specified query string parameter, the Engagement Analytics engine records the specified event.

Note
The default Engagement Analytics configuration maps the URL query string parameter named `sc_rss` to the event named RSS.

2.1.6 Campaign Categories

A campaign category is a marketing effort to promote awareness of a product, service, or other offering through one or more media channels. You can associate each visit with one or more campaign categories. Engagement Analytics users can filter visit data by campaign. For more information about working with campaigns and campaign categories see the *Marketing Operations Cookbook*.

2.1.7 Reverse DNS Lookups

A reverse DNS lookup provides a human-readable name of the server at the specified address, which may be used to find information about the owner of an IP address.

2.1.8 GeoIP Lookups

A GeoIP lookup provides information about the location and owner of an IP address beyond that provided by a reverse DNS lookup. To include GeoIP information in reports, the Engagement Analytics engine uses an external service that performs GeoIP lookups.

Note
Sitecore customers must contract with a third-party such as MaxMind to obtain GeoIP lookup service. For more information about MaxMind, see the section *MaxMind GeoIP Lookup Provider Settings*. This product includes GeoLite data created by MaxMind (http://www.maxmind.com/). For more information about MaxMind, see http://sdn.sitecore.net/Products/OMS/MaxMind.aspx.

2.1.9 Automatic Robot Detection

The Engagement Analytics engine can automatically detect robots such as search engines, classifying their visits separately from those of actual users. Engagement Analytics will by default drop robot sessions. This can be disabled if there are visitors incorrectly identified as robots.

Sitecore uses a number of criteria to attempt to identify robots. A robot is likely to place a number of requests in a short period of time, is unlikely to support cookies, and is unlikely to request CSS files and media referenced by content. If needed, users can reclassify robot visits as live visits.

2.1.10 Engagement Analytics Email Distribution

To configure Engagement Analytics report distribution by email, in the Content Editor, edit the Schedule field in the Data section of the `/Sitecore/System/Settings/Analytics/Reports/Reports` item. If this field contains a value, it must be an XML document with the root element `<schedule>`. Each child of the `<schedule>` element must be a `<report>` element. Each `<report>` element designates the properties of a single email report.

For example, with this XML in the Schedule field:

```
<schedule>
  <report
    reportitem="/sitecore/system/Settings/Analytics/Reports/Reports/Marketing/Metrics"
    recipients="alias@domain.tld"
    recurrence="everyday"
    title="Metrics"
    days="1"
    format="html" />
  <!-- additional <report> elements -->
</schedule>
```

The attributes of the `<report>` element described in the following table are mandatory:

Attribute	Description
`reportitem`	The full path of the report definition item.
`recipients`	Email addresses separated by commas.
`Recurrence`	Frequency of distribution (`everyday`, `weekdays`, `Mondays`, `Tuesdays`, `Wednesdays`, `Thursdays`, `Fridays`, `Saturdays`, `Sundays`, `monthly`, or `biweekly`). Separate multiple values with commas (“,”). This value is not case sensitive.
`Title`	Email message title.
`Days`	Number of days of analytics data to include in the report.
`Format`	Report format (`html`, `pdf`, `png`, `rtf`, `word2007`, or `excel2007`).

The `/configuration/sitecore/scheduling/agent` element in `web.config` with `type` `Sitecore.Analytics.Tasks.EmailReportsTask` distributes analytics reports by email. This agent runs hourly and checks the value in the **Last Run** field in the `/Sitecore/System/Settings/Analytics/Reports/Reports` item in order to run report

processing once each day, shortly after midnight. The **Last Run** field is updated after report processing is completed.

Note
Sitecore uses the SMTP server that is specified in the `web.config` file in the `MailServer` setting.

Note
You can also distribute Engagement Analytics reports by email using the Sitecore user interface and APIs.

Note
The `Sitecore.Analytics.Tasks.EmailReportsTask` task is configured by default to use the *master* database. To ensure that the task will execute in Content Delivery environments where the *master* database is absent, you must change the `DatabaseName` setting to `web`.

2.2 Multivariate Tests

You can use the Engagement Analytics engine to apply multivariate testing You can use multivariate tests to dynamically set the data source of each presentation control.

2.2.1 Multivariate Test Definitions and Multivariate Test Variables

Each multivariate test consists of a multivariate test definition item containing one or more multivariate test variables (the TestSet). Each multivariate test variable specifies an item to apply as the data source of the presentation control when the system applies that variable. The multivariate test definition item specifies a strategy for the system to select which of the multivariate test variables to apply as the data source for the rendering. Layout details allow the user to select a multivariate test to apply to each presentation control. For more information about configuring multivariate tests and test variables see the *Marketing Operations Cookbook*.

2.3 Engagement Analytics Architecture

The Engagement Analytics engine adds analytics processing logic to each page request.

2.3.1 The Engagement Analytics Data Queue

Each page request can generate a number of page events and user profile information. To reduce performance impact while rendering pages, Sitecore engagement analytics engine writes this data to an in memory Data Set for the visitor. Once the visitor data has reached a specified size or reached a specified time limit, data is moved to an in memory shared DataSet. Once the shared DataSet reaches a specified size or time limit, bulk database operations commit the data to persistent database storage.

For instructions to configure the settings for the size and time limits, see the section *The Analytics.MaxQueueSize Settings* in this document.

Important
Configure the queue size to prevent overwhelming the solution infrastructure at peak periods, or to reduce the harm of aggressive, unrecognized search engines.

Chapter 3

Engagement Analytics Configuration

This chapter provides instructions to configure the Engagement Analytics engine, including databases, settings, and providers.

This chapter contains the following sections:

- The Analytics Database
- The Analytics Client Roles
- The Analytics Configuration Settings
- The Analytics Provider
- The GeoIP Lookup Provider
- The Site Definition enableAnalytics Attribute
- Using Separate Databases for Collecting Data and Reporting in Sitecore DMS

3.1 The Analytics Database

The `connectionString` attribute of the `/connectionStrings/add` element in `/App_Config/ConnectionStrings.config` with `name=analytics` specifies the database used on the website to collect the analytics data. Multiple Sitecore instances can access the same analytics database. For example, multiple content delivery instances behind a load balancer might populate an analytics database accessed by a separate content management server to generate reports.

The `connectionString` attribute of the `/connectionStrings/add` element in `/App_Config/ConnectionStrings.config` with `name=reporting` specifies the database for use by reporting applications. Stimulsoft Reports, the Executive Dashboard, EAS Supervisor and EAS Monitor will use the connection string for the reporting database if it is defined.

Warning
Sitecore supports Microsoft SQL Server for analytics features.

3.1.1 Disabling the Analytics Database

The `Sitecore.Analytics.config` file contains a setting that allows you to disable the *Analytics* database. This configuration file is stored in the `\wwwroot\SitecoreInstallation\Website\App_Config\Include` folder.

The setting is: `<setting name="Analytics.DisableDatabase" value="false" />`

The default value is false.

If you disable the *Analytics* database, no information is read from or written to the database.

Runtime Personalization

However, if you disable the *Analytics* database, you can still use Sitecore to perform runtime personalization — personalization that is based on the behavior of the visitor during the current session.

There are a number of personalization rules that are not based on data from the database. For example, rules that depend on devices, goals, events, and traffic type to name just a few.

Needless to say, you will not be able to implement any personalization rules that depend on historical data.

3.2 The Analytics Client Roles

The `Sitecore\Analytics` security roles control access to Engagement Analytics features in the Sitecore user interface. These roles include Client Profiling, Maintaining, Personalization and Reporting.

For more information about the Sitecore client security roles, see the *Client Configuration Cookbook*.

3.3 The Analytics Configuration Settings

Comments above each `/configuration/sitecore/settings/setting` element in the configuration file provide possible values for each setting.

Important
For an important note regarding `web.config`, see the section Engagement Analytics Features and ConceptsEngagement Analytics Features and Concepts.

The Analytics.AutoDetectBots Setting

The `value` attribute of the `/configuration/sitecore/settings/setting` element in `web.config` with `name Analytics.AutoDetectBots` specifies whether the analytics engine attempts to classify search engines and other web robots automatically. For more information about automatic robot detection, see the section *Automatic Robot Detection*.

The Analytics.BulkCopyBatchSize Setting

The `value` attribute of the `/configuration/sitecore/settings/setting` element in `web.config` with `name Analytics.BulkCopyBatchSize` specifies the number of rows to batch in a bulk copy operation when transferring analytics data from memory to persistent database storage.

The Analytics.CampaignQueryStringKey Setting

The `value` attribute of the `/configuration/sitecore/settings/setting` element in `web.config` with `name Analytics.CampaignQueryStringKey` specifies the name of the URL query string parameter used by the Engagement Analytics engine to trigger a campaign. For more information about campaigns, see the section *Campaign*.

The Analytics.DisableDatabase Setting

The `value` attribute of the `/configuration/sitecore/settings/setting` element in `web.config` with `name Analytics.DisableDatabase` allows you to disable reading and writing to the Analytics database.

For more information, see the section *Disabling the Analytics Database*.

The Analytics.EMailFromAddress Setting

The `value` attribute of the `/configuration/sitecore/settings/setting` element in `web.config` with `name Analytics. EMailFromAddress` specifies the email address to use as the sender for messages generated by the Engagement Analytics engine. For more information about email distribution, see the section Engagement Analytics Email DistributionEngagement Analytics Email Distribution.

The Analytics.Enabled Setting

The `value` attribute of the `/configuration/sitecore/settings/setting` element in `web.config` with `name Analytics.Enabled` controls whether Sitecore stores information about incoming HTTP requests. To improve performance in environments that do not use analytics, set `Analytics.Enabled` to `false`.

The Analytics.EventQueryStringKey Setting

The `value` attribute of the `/configuration/sitecore/settings/setting` element in `web.config` with `name Analytics.EventQueryStringKey` specifies the name of the URL query string parameter used by the Engagement Analytics engine to trigger an event. For more information about events, see the section Engagement Analytics Page Events.The Engagement Analytics Data Queue

The Analytics.GeoIps.CacheSize Setting

The `value` attribute of the `/configuration/sitecore/settings/setting` element in `web.config` with `name` `Analytics.GeoIps.CacheSize` specifies the number of cached items in the GeoIP lookup manager.

The Analytics.LogLevel Setting

The `value` attribute of the `/configuration/sitecore/settings/setting` element in `web.config` with `name` `Analytics.LogLevel` specifies the threshold for logging activity. By default, the Engagement Analytics engine logs only errors and exceptions.

The Analytics.LongRunningRequest Setting

The `value` attribute of the `/configuration/sitecore/settings/setting` element in `web.config` with `name` `Analytics.LongRunningRequest` specifies the number of milliseconds to allow for Engagement Analytics operations. If an operation exceeds this threshold, that operation will appear as an error in the Sitecore log and Engagement Analytics reports.

The Analytics.MaxCachedRows Setting

The `value` attribute of the `/configuration/sitecore/settings/setting` element in `web.config` with `name` `Analytics.MaxCachedRows` specifies how many rows of data are kept in memory for each of the definition tables. When this value is exceeded, half of the rows are discarded from memory.

The Analytics.MaxQueueSize Setting

The `value` attribute of the `/configuration/sitecore/settings/setting` element in `web.config` with `name` `Analytics.MaxQueueSize` controls the maximum size of the data queue. For more information about the data queue, see the section The Engagement Analytics Data Queue. The default value is now 10,000.

Warning
The size of the Engagement Analytics data queue cannot exceed memory available to ASP.NET. Reaching the data queue limit does not always indicate an inadequate queue size. Reaching the queue limit may indicate inadequate hardware to service peak load, an aggressive robot, or other factors.

The Analytics.PerformLookup Setting

The `value` attribute of the `/configuration/sitecore/settings/setting` element in `web.config` with `name` `Analytics.PerformLookup` specifies whether the instance performs reverse DNS and GeoIP lookups. Only one of the Sitecore instances connected to an Analytics database should perform lookups.

Important
Most GeoIP lookup providers, including the MaxMind provider, require Internet access to perform GeoIP lookups. Set `Analytics.PerformLookup` to `false` in environments without Internet access.

The Analytics.PerformLookup.Interval Setting

The `value` attribute of the `/configuration/sitecore/settings/setting` element in `web.config` with `name` `Analytics.PerformLookup.Interval` controls the interval between lookup operations.

The Analytics.ReverseDnsLookupTimeout Setting

The `value` attribute of the `/configuration/sitecore/settings/setting` element in `web.config` with `name` `Analytics.ReverseDnsLookupTimeout` specifies a number of milliseconds to allow for reverse DNS lookup operations.

The Analytics.Robots.IgnoreRobots Setting

The `value` attribute of the `/configuration/sitecore/settings/setting` element in `web.config` with `name` `Analytics.Robots.IgnoreRobots`, when set to `true`, specifies that visit data from requests detected as robots will not be written to the database.

The Analytics.Robots.IgnoreAutoRobots Setting

The `value` attribute of the `/configuration/sitecore/settings/setting` element in `web.config` with `name` `Analytics.IgnoreAutoRobots`, when set to `true`, specifies that visit data from requests detected as auto robots will not be written to the database.

The Analytics.Sampling.Percentage Setting

The `value` attribute of the `/configuration/sitecore/settings/setting` element in `web.config` with `name` `Analytics.Sampling.Percentage` specifies the percentage of visits that should be collected in the Analytics database. The default is 100, all data for all visits will be collected.

The Analytics.TrackerChanges.FlushInterval Setting

The `value` attribute of the `/configuration/sitecore/settings/setting` element in `web.config` with `name` `Analytics.TrackerChanges.FlushInterval` specifies how frequently analytics tracking data is flushed from memory to the database.

The Analytics.TrackerChanges.MaxRows Setting

The `value` attribute of the `/configuration/sitecore/settings/setting` element in `web.config` with `name` `Analytics.TrackerChanges.MaxRows` specifies how many rows of analytics tracking data is held in memory before the data is flushed from memory to the database.

The Analytics.TrackerChanges.Sleep Setting

The `value` attribute of the `/configuration/sitecore/settings/setting` element in `web.config` with `name` `Analytics.TrackerChanges.Sleep` specifies the tracker changes queue inactive period.

The Analytics.UpdateBatchSize Setting

The `value` attribute of the `/configuration/sitecore/settings/setting` element in `web.config` with `name` `Analytics.UpdateBatchSize` specifies the number of SQL requests to include in a batch operation.

The Analytics.UseBulkCopy Setting

The `value` attribute of the `/configuration/sitecore/settings/setting` element in `web.config` with `name` `Analytics.UseBulkCopy` specifies whether or not a bulk copy operation should be used to insert rows into the database.

The Analytics.Automation.BatchSize Setting

The `value` attribute of the `/configuration/sitecore/settings/setting` element in `web.config` with `name` `Analytics.Automation.BatchSize` specifies the number of records to process in a batch.

The Analytics.Automation.Interval Setting

The `value` attribute of the `/configuration/sitecore/settings/setting` element in `web.config` with `name` `Analytics.Automation.Interval` specifies how frequently the workflow worker process runs.

The Analytics.Automation.MachineName Setting

The `value` attribute of the `/configuration/sitecore/settings/setting` element in `web.config` with `name` `Analytics.Automation.MachineName` specifies the name of the server that the workflow engine runs on.

The Analytics.Automation.Threads Setting

The `value` attribute of the `/configuration/sitecore/settings/setting` element in `web.config` with `name` `Analytics.Automation.Threads` specifies the number of threads used by the workflow engine.

3.3.1 Additional Engagement Analytics Settings

The configuration file entries described in this section are not included in the `Sitecore.Analytics.config` file that is provided during DMS installation. You must add these entries yourself to enable the related functionality.

The Analytics.Patterns.MinimalProfileScoreCount Setting

If you are using Pattern Cards you must add the following line to `Sitecore.Analytics.config` file, `configuration/sitecore/settings` section:

```
        <setting name= "Analytics.Patterns.MinimalProfileScoreCount" value= "3" />
```

The `value` attribute of this setting specifies how many pages need to be accessed during a visit before pattern matching behavior begins. For example, if the `value= "3"` then pattern matching will begin when the number of pages visited is greater than 3.

For more information about Profiles, Profile Cards and Pattern Cards see the *Sitecore Marketing Operations Cookbook*.

3.4 The Analytics Provider

To update the default Analytics provider to store Engagement Analytics data in a system other than the default Sitecore Analytics database, implement a custom Analytics provider class that provides the same signature as the default Analytics provider.

Then, either update the `type` attribute of the `/configuration/sitecore/dataAdapterManager/providers/add` element in `Sitecore.Analytics.config` with `name="default"` to reference your class, or add an additional `<add>` element within the `<providers>` element, give it a unique `name` and update the `type` attribute to reference your provider, and set the `defaultProvider` attribute of the `<dataAdapterManager>` element to that `name`.

```
<dataAdapterManager defaultProvider="default">
      <providers>
        <clear/>
        <add name="default" type="Sitecore.Analytics.Data.DataAccess.DataAdapters.
          Sql.SqlServer.SqlServerDataAdapterProvider, Sitecore.Analytics"/>
      </providers>
</dataAdapterManager>
```

Important
Your Analytics provider must provide the same signature as the default Analytics provider.

3.5 The GeoIP Lookup Provider

The `type` attribute of the `/configuration/sitecore/lookupManager/providers/add` element in `web.config` with `name default` specifies the GeoIP lookup provider.

3.5.1 How to Implement a GeoIP Lookup Provider

To implement a custom GeoIP lookup provider:

1. In the Visual Studio web application project, create a class that inherits from the `Sitecore.Analytics.Lookups.LookupProviderBase` class.
2. In the new class, implement the `GetInformationByIp()` method.
3. In the `/configuration/sitecore/lookupManager/providers/add` element in the `web.config` file, update the `type` attribute to reference the new class.

3.5.2 MaxMind GeoIP Lookup Provider Settings

To use the MaxMind GeoIP lookup provider, configure the settings described in this section, and then set the `type` attribute of the `/configuration/sitecore/lookupManager/providers/add` element in `web.config` with `name default` set to the following value:

```
Sitecore.Analytics.Lookups.MaxMindProvider, Sitecore.Analytics
```

While Engagement Analytics comes with a number of free lookups for testing and implementation convenience, and to ensure that you can utilize CEP capabilities from day one, the free batch of lookups will expire and additional lookups must be purchased from MaxMind.

For information on signing up with MaxMind and purchasing more lookups, refer to the following page: http://sdn.sitecore.net/Products/OMS/MaxMind.aspx.

Important
If you use the paid MaxMind GeoIP lookup service, you must change the `MaxMind.Url` setting to the URL provided to you by MaxMind. For more information about the `MaxMind.Url` setting, see the section *The MaxMind.Url Setting*.

For assistance on any issues about using the GeoIP web service within Sitecore CMS, please contact MaxMind's support department at support@maxmind.com or use the chat service that is available on www.maxmind.com between 9:30 - 17:00 Eastern Time, Monday to Friday.

All messages received during normal business hours receive a timely, same day response. All other messages receive a response during the next business day.

Note
You can use your MaxMind-Sitecore license key to purchase additional queries at www.maxmind.com/app/sitecore.

The MaxMind.Encoding Setting

The `value` attribute of the `/configuration/sitecore/settings/setting` element in `web.config` with `name MaxMind.Encoding` controls the page encoding used by the MaxMind GeoIP lookup provider.

The MaxMind.Format Setting

The `value` attribute of the `/configuration/sitecore/settings/setting` element in `web.config` with `name MaxMind.Format` controls the format used by the MaxMind GeoIP lookup provider.

The MaxMind.SecurityToken Setting

The `value` attribute of the `/configuration/sitecore/settings/setting` element in `web.config` with `name` `MaxMind.SecurityToken` controls the security used by the MaxMind GeoIP lookup provider.

The MaxMind.Url Setting

The `value` attribute of the `/configuration/sitecore/settings/setting` element in `web.config` with `name` `MaxMind.Url` controls the URL used by the MaxMind GeoIP lookup provider.

3.6 The Site Definition enableAnalytics Attribute

To enable or disable capturing of Engagement Analytics data for each web site, you can set the `enableAnalytics` attribute of the appropriate `/configuration/sitecore/sites/site` element in `web.config` to `true` or `false`.

Tip
If you separate content management from content delivery, to prevent capturing analytics data in the content management environment, in the content management environment, set the `enableAnalytics` attribute to `false` for all logical sites.

3.7 Using Separate Databases for Collecting Data and Reporting in Sitecore DMS

With the release of Sitecore 6.6, Sitecore supports the use of separate databases for collecting data and for reporting. This implementation improves the performance of both writing data to the database and extracting data from the database. The collection database — *Sitecore_Analytics* — has been optimized for writing data and for the limited extraction of data. The reporting database is optimized for the mass extraction of data.

Further performance improvements can be gained by deploying the collection and reporting databases on separate physical servers.

You can improve performance even further by setting up a dedicated automation database that is shared between the Content Delivery (CD) and the Content Management (CM) environments and distributes the load between the physical SQL servers.

3.7.1 Transferring Data

Data transfer between the collection and reporting database is performed by an SSIS package which is run periodically by a SQL Server Agent Scheduled Job.

The SSIS package is called *Sitecore DMS Data Transfer Package.* You can download the latest version of this package from the Sitecore 6.6 section of the SDN.

The SSIS package allows you to configure the following parameters:

Parameter	Description
SourceConnection (Data Source)	The connection string for the source database — collection. You can change this parameter when you set up the scheduled job that runs the package.
DestinationConnection (Data Source)	The connection string for the destination database — reporting. You can change this parameter when you set up the scheduled job that runs the package.
bufferTime (Variable)	The delay in seconds between the time a record appears in the collection database (such as when a visit is registered) and when this record is copied to the reporting database. You can change this parameter when you set up the scheduled job that runs the package. Default value: 3600 seconds.

The following schedule sets up the package so that it is run every night at 23.00. This prepares the data for another background job that is executed in the DMS (Reporting) database every night at 23.30 and updates the summary information in the Executive Insight Dashboard and in the Engagement Analytics reports.

Schedule for DMS background jobs:

Order	Schedule	Name
1	Every night at 23:00	Transfer Data from *Collection* to *Reporting* database
2	Every night at 23:30	Refresh Summary Views in the *Reporting* database Default value: runs once a day at 23:30.

The database administrator can change the schedules of both jobs, but they must be executed in this order. You must also allow enough time for the first job to finish before the second job starts, otherwise the performance of the second job may suffer.

The second DMS background job refreshes the summary information in the Executive Insight Dashboard and in the Engagement Analytics reports after you update the data in the *Reporting* database.

You specify the schedule for this background job in the `Sitecore.Analytics.Config` file, in the scheduling section:

```
<agent type="Sitecore.Analytics.Tasks.UpdateReportsSummaryTask, Sitecore.Analytics"
method="Run" interval="00:00:00">
<UpdateReportsSummaryAt>23:30:00</UpdateReportsSummaryAt>
</agent>
```

3.7.2 Configuring Sitecore to use Separate Databases for Collecting and Reporting

The following instructions assume that the:

- Existing DMS database is called *Sitecore_Analytics* and is used as the collection database.
- New Reporting DMS database is called *Sitecore_Reporting*.
- New Automations EAS database is called *Sitecore_Automations*.

Overview

To configure Sitecore DMS to use separate databases for collecting data and reporting, you must:

1. Make a copy of your *Sitecore_Analytics* database and call it *Sitecore_Reporting*.
2. Deploy the *Sitecore DMS Data Transfer Package* on SQL Server.
3. Start SQL Server Management Studio, connect to the *Sitecore_Analytics* database and execute `DatabaseAddon.sql` from *Sitecore DMS Data Transfer Package*.

   This prepares the *Sitecore_Analytics* database for the SSIS package.
4. Set up a scheduled job to transfer data from *Sitecore_Analytics* to *Sitecore_Reporting* database.
5. Reconfigure the Sitecore CMS instances to use the following connection strings:
   - `<add name="analytics" connectionString="…Sitecore_Analytics"/>`
   - `<add name="reporting" connectionString="…Sitecore_Reporting"/>`
6. If necessary, set up a dedicated automations database.

The following sections describe the more complicated steps in detail.

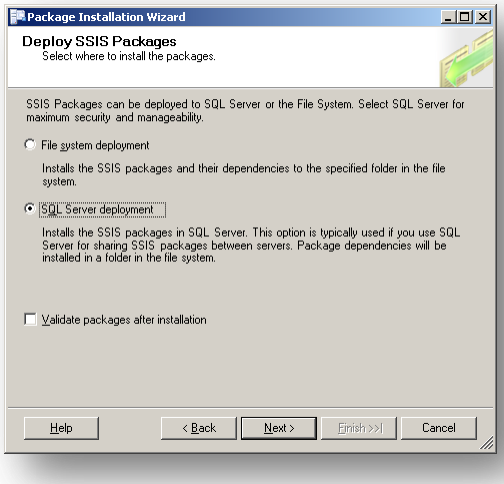
Deploying the Sitecore DMS Data Transfer Package on SQL Server

1. Unzip the *Sitecore DMS Data Transfer Package*.

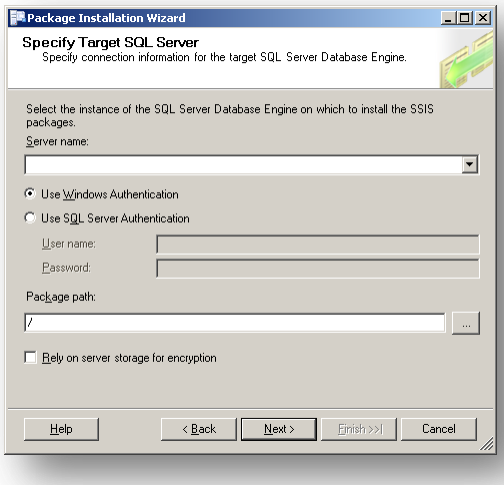
   Place it in a folder on the same server as the SQL Server instance where the *Sitecore_Reporting* database is deployed.
2. Double-click `[MoveAnalyticsData.SSISDeploymentManifest]` and the **Package Installation Wizard** opens.

   This wizard will guide you through the installation process.

3. In the **Deploy SSIS Packages** window, select **SQL Server deployment**.

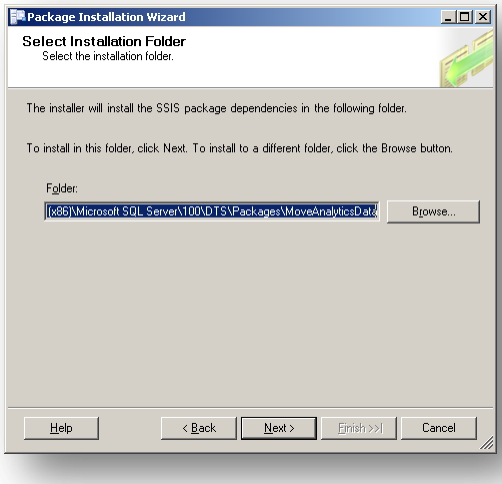


4. In the **Specify Target SQL Server** window, enter the server name and credentials that you want to use.



5. In the **Package path** field specify / as the path.

6. In the **Select Installation Folder** window, select the installation path that you want to use for the SSIS package.



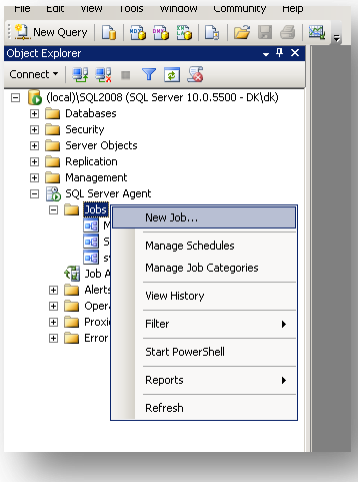
7. Complete the installation wizard and install the package.

Setting up a Scheduled Job

After you have installed the SSIS package, you can set up a scheduled job to transfer data from the *Sitecore_Analytics* database to the *Sitecore_Reporting* database.

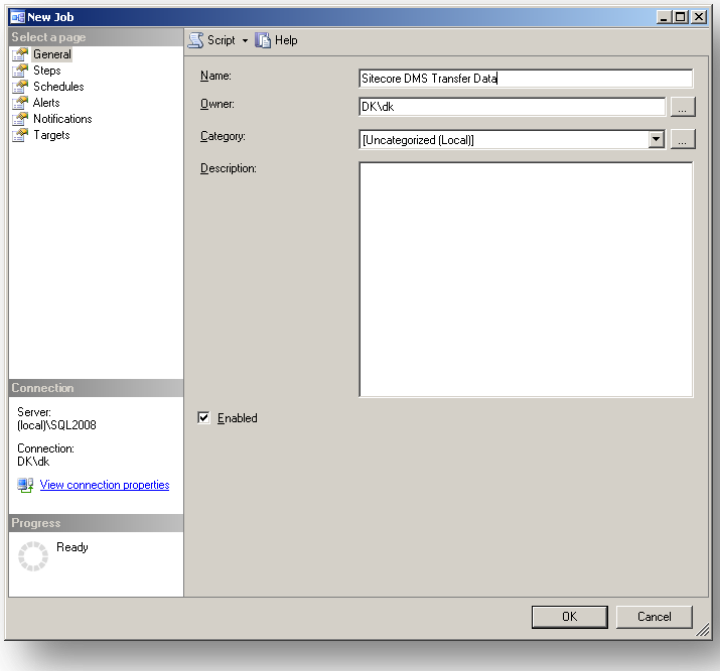
To set up a scheduled job:

1. Open SQL Server Management Studio and connect to the SQL Server instance where the *Sitecore_Reporting* database is deployed.

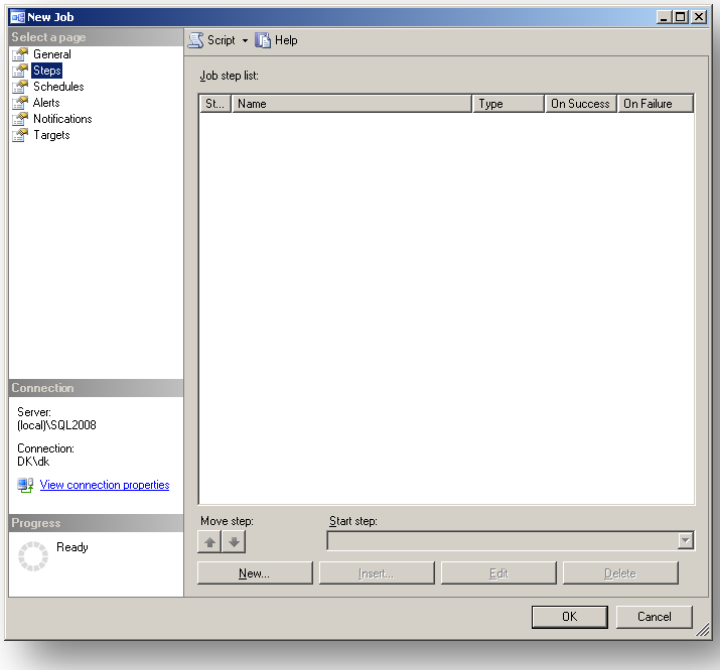


2. In the **Object Explorer**, expand the *SQL Server Agent* node.

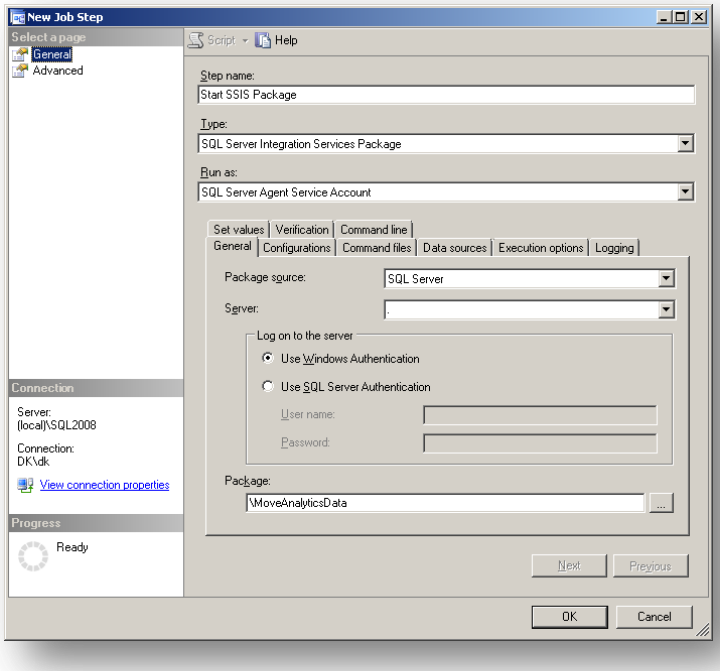
3. Right-click the *Jobs* node and then click select **New Job…**



4. In the **New Job** dialog box, give the job a unique name.
5. In the **Select a page** panel, click *Steps.*

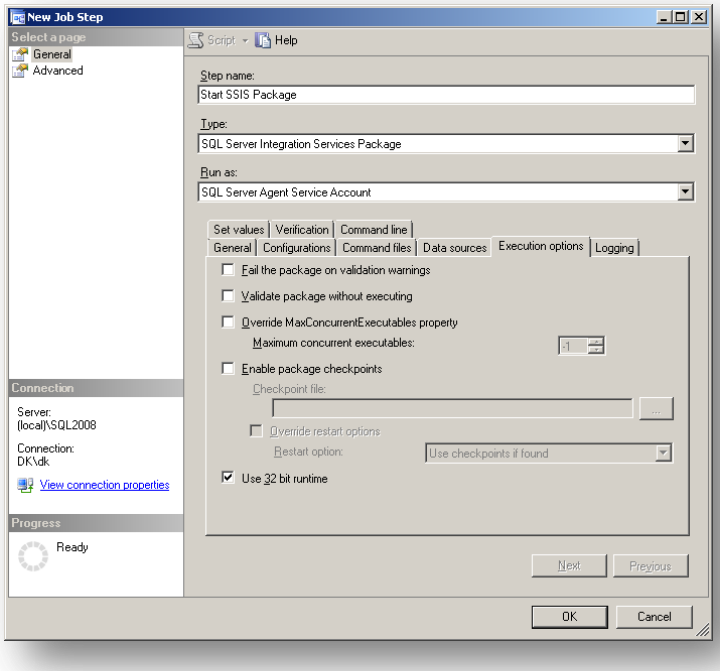


6. Click **New…** to create a new step.

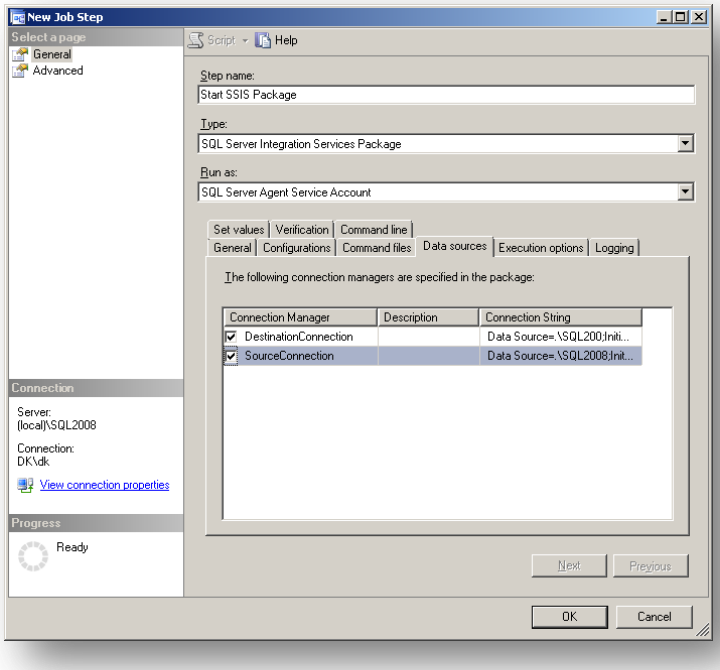


7. In the **New Job Step** dialog box, in the **Step name** field, give the step a unique name.
8. In the **Type** field, select *SQL Server Integration Services Package*.
9. On the **General** tab, in the **Package source** field, select *SQL Server*.
10. In the **Server** field, select the instance of SQL Server instance where the package is deployed.
11. In the **Log on to server** section, select the type of authentication that you want to use and enter the appropriate credentials for the SQL Server instance.
12. In the **Package** field, click the browse button and select the package.

13. Click the **Execution options** tab.



14. Select the **Use 32 bit runtime** checkbox.
15. Click the **Data Sources** tab.

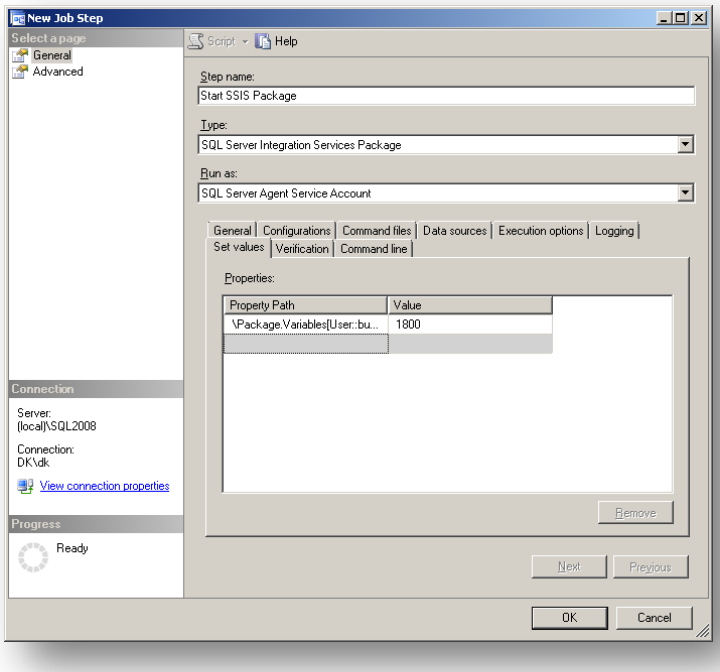


16. Select the *DestinationConnection* and the *SourceConnection* connection managers.
17. Specify the connection strings for the destination database (*Sitecore_Reporting*) and the source database (*Sitecore_Analytics*).

The connection strings should have the following format:

*Data Source=**<server name>**;Initial Catalog=**<database name>**;Provider=SQLNCLI10.1;Integrated Security=SSPI;Auto Translate=false;Application Name=SSIS-Transfer DMS;*

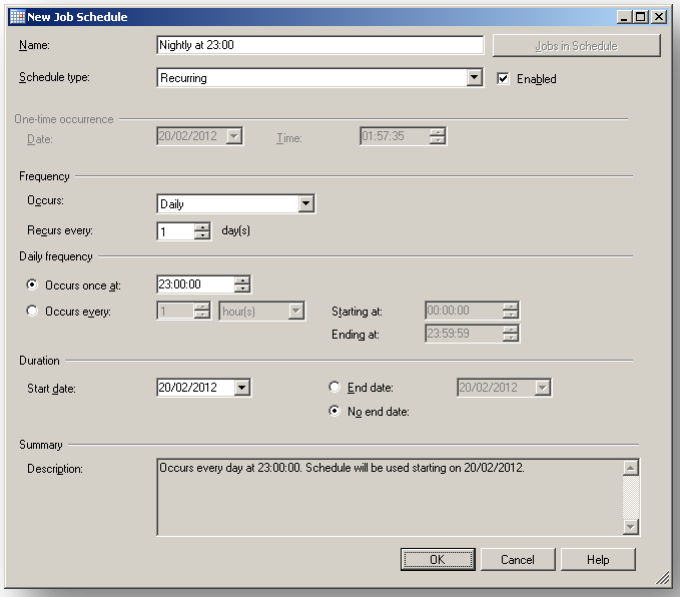
18. Replace the values in brackets with the real names of the server and the database.
19. If you need to change the *bufferTime* variable, click the **Set values** tab.



The *bufferTime* variable defines the delay in seconds between the time a record appears in the collection database (such as when a visit is registered) and when this record is copied to the reporting database.

20. In the **Properties** field, in the **Property Path** column, enter `\Package.Variables[User::bufferTime].Properties[Value].`
21. In the **Value** column, enter the value that you think is appropriate.
22. Click **OK** to return to the **New Job** dialog box.

23. In the **New Job** dialog box, in the **Select a page** panel, click *Schedule* and then click **New** to open the **New Job Schedule** dialog box.



24. Set up a schedule to run every day at, for example, 23:00.
25. Click **OK** to return to the **New Job** dialog.
26. Click **OK** to create the job.

Setting up a Dedicated Automations Database

To set up a dedicated automations database:

1. Make a copy of the *Sitecore_Analytics* database and call it *Sitecore_Automations*.
2. Configure the connection strings in both the CM and the CD environments:

   ```
   <add name="automations" connectionString="…Sitecore_Automations" >
   ```

3. In the *Sitecore_Analytics* database, to clear the *AutomationStates* table, execute the following statement:

   ```
   DELETE FROM [AutomationStates];
   ```

3.8 Setting Up a SQL Server Agent to Refresh the Report Summary Data in the Sitecore_Analytics Database

The information in this section is applies to Sitecore 6.6 or later.

The `Sitecore.Analytics.config` file contains a setting called `UpdateReportsSummaryTask` that specifies that a SQL Agent should refresh the report summary data in the *Sitecore_Analytics* database. This setting is the default method for automatically refreshing the views.

For more information about the `UpdateReportsSummaryTask` setting, see the section *Using Separate Databases for Collecting Data and Reporting in Sitecore DMS*.

However you can also configure a SQL Server Agent job to perform this task. This SQL Server Agent job is a more manageable, reliable, and configurable method for automatically refreshing the views.

This SQL Server Agent job ensures that the latest data is available in your reports and in the dashboards.

Prerequisites

- SQL Server Standard Edition or higher.
- Download the `DMS660_Job.sql` script from the Sitecore 6.6 section of the SDN.

Note
SQL Server Express Edition does not support SQL Server Agent.

Configuration

1. To create the update job, in your *Sitecore_Analytics* database, execute `DMS660_Job.sql`.

   This creates a job called *Refresh Summary Views (<name of your Sitecore_Analytics database>)*.
2. Adjust the job schedule to meet the requirements of your solution — specify when the job should be executed and the interval that SQL Server should wait for the job to complete before shutting down.
3. Run the job and verify that it executes correctly.
4. Open the `Sitecore.Analytics.config` file.

   This file is stored in the `C:\inetpub\wwwroot\Your Sitecore Installation\Website\App_Config\Include` folder.
5. Comment out the definition of the agent type — `Sitecore.Analytics.Tasks.UpdateReportsSummaryTask`.

```
        <scheduling>
              <agent type="Sitecore.Analytics.Tasks.EmailReportsTask, Sitecore.Analytics"
method="Run" interval="1:00:00">
                <DatabaseName>master</DatabaseName>
              </agent>
              <agent type="Sitecore.Analytics.Tasks.SubscriptionTask, Sitecore.Analytics"
method="Run" interval="00:15:00">
                <Minutes>15</Minutes>
              </agent>
              <!--<agent type="Sitecore.Analytics.Tasks.UpdateReportsSummaryTask,
Sitecore.Analytics" method="Run" interval="00:30:00">
                <UpdateReportsSummaryAt>23:30:00</UpdateReportsSummaryAt>
              </agent>-->
            </scheduling>
```

This new SQL Server Agent job and the agent defined in the `Sitecore.Analytics.config` file perform the same task.

If you are running on SQL Server Express, you must use the agent defined in the `Sitecore.Analytics.config` file.

3.9 Executive Dashboard Configuration Files

The Sitecore dashboards use several separate configuration files:

XML File	Used by
DefaultSettings.configl	Executive Insight Dashboard
Configuration.config	Executive Insight Dashboard
CampaignCategoryDefaultSettings.config	Single Campaign Category Dashboard
SingleCampaignDefaultSettings.config	Single Campaign Dashboard
SingleTraficTypeDefaultSettings.config	Single Traffic Type Dashboard

Each of these files has the same format but they contain different settings.

The `DefaultSettings.config` file contains several settings that you can use to specify the properties that are sent to the Executive Insight Dashboard. When the dashboard application asks the Integration web handler to return the default settings, it sends the path to the file — defined in the `Configuration.config` — together with the appropriate query string parameter.

There are three query string parameters:

```
• <SingleCampaignDefautSettingUrl value="/data/SingleCampaignDefaultSettings.config"/>
• <SingleTrafficTypeDefautSettingUrl
    value="/data/SingleTrafficTypeDefaultSettings.config"/>
• <CampaignCategoryDefautSettingUrl
    value="/data/CampaignCategoryDefaultSettings.config"/>
```

Sitecore developers can use these parameters to override the default setting values or redirect to a different default settings file.

The settings in these config files allow you to define various features in the dashboards including:

- The time range of the data that the dashboards should use.
- The websites that the dashboards should display data for.
- The default website and the default language that the dashboards should use.
- The dashboards that should be available.
- The default dashboard.

The `configuraton.config` file is stored in the `\Website\sitecore\shell\Applications\Reports\Dashboard` folder of your installation.

The settings in the `configuraton.config` file include:

Setting	Defines
`<ChartPageSize value="20"/>`	The number of items that are displayed in the vertical axis of the charts when the scrollbar is visible.
`<ChartItemsTopNumber value="100"/>`	The number of entries that are displayed in a chart. In this example, the top 100 entries are displayed.
`<MinimumVisitsFilter value="1" />`	The minimum number of visits that are displayed in the charts. Anything less than this threshold is regarded as being statistically insignificant.
`<DataTimeFrom value="20080101"/>`	The start date from which data is retrieved.

Setting	Defines
`<!--<DataIntervalDays value="365"/>-->`	The number of days back in time from UTC.Now for which data is retrieved.

Chapter 4

The Sitecore.Analytics API

This chapter describes the Sitecore.Analytics Application Programming Interface (API).

This chapter contains the following sections:

- Sitecore.Analytics API Overview

4.1 Sitecore.Analytics API Overview

To use the Sitecore.Analytics APIs, in the Visual Studio project, add a reference to the `Sitecore.Analytics.dll` assembly in the `/bin` folder within the document root of the Sitecore solution website. For more information about the Engagement Analytics API see the *Sitecore Engagement Analytics API Cookbook*.

Warning
In the Visual Studio project, set the *Copy Local* property of all references in the project to assemblies in the `/bin` folder to `False`. If you do not set the *Copy Local* property of a referenced assembly to `False`, Visual Studio can delete assemblies from the `/bin` folder, causing Sitecore to fail.

4.1.1 Sitecore.Analytics.Tracker

The `Sitecore.Analytics.Tracker` static class provides access to the `Tracker.DataContext property`. The `Sitecore.Analytics.Tracker.IsActive` property indicates whether analytics are enabled for the current request.

Sitecore.Analytics.Tracker.CurrentPage

The `Sitecore.Analytics.Tracker.CurrentPage` class represents a page in a visit. You can associate events with the previous, current, and next pages in the visit.

Sitecore.Analytics.Tracker.CurrentVisit

The `Sitecore.Analytics.Tracker.CurrentVisit` class represents activity during the current visit.

4.1.2 Sitecore.Analytics.TrackerDataContext

The `Sitecore.Analytics.TrackerDataContext` provides information about the current visit including PageEvents, Pages, Campaigns, GeoIps, and Tags.

Sitecore.Analytics.TrackerDataContent.Tags

You can use the `Sitecore.Analytics.TrackerDataContext.Tags` collection to access Engagement Analytics tags. The tags collection consists of name-value pairs that you can assign programmatically.

4.1.3 The VisitorIdentification Web Control

The `VisitorIdentification` web control helps the Sitecore analytics engine identify robots. The `VisitorIdentification` web control generates an HTML `<link>` element that references an empty CSS resource. The Engagement Analytics engine uses this request to help differentiate robots from actual users. For more information about automatic robot detection, see the section *Automatic Robot Detection.*

Important
To assist the Engagement Analytics engine in robot identification, include the `VisitorIdentification` web control in the HTML `<head>` section of all of your layouts. For example:

```
<sc:VisitorIdentification runat="server" />
```

Chapter 5

Engagement Analytics Troubleshooting

This chapter provides instructions to troubleshoot issues with the Engagement Analytics engine.

This chapter contains the following sections:

- Engagement Analytics Troubleshooting Overview

5.1 Engagement Analytics Troubleshooting Overview

Use the information in the following sections to troubleshoot issues with the Engagement Analytics engine.

5.1.1 Sitecore Logs

Always begin troubleshooting by investigating the Sitecore log files in the directory specified by the `value` attribute `/configuration/sitecore/settings/setting` element in `web.config` with `name LogFolder`.

5.1.2 Engagement Analytics Control Panel

The **Engagement Analytics** section of the **Control Panel** in the Sitecore desktop provides information about reverse DNS and GeoIP lookup success and failure.

To determine if Sitecore is able to perform reverse DNS and GeoIP lookups:

1. In the Sitecore Desktop, click the **Sitecore** button. The **Sitecore** menu appears.
2. In the **Sitecore** menu, click **Control Panel**. The **Control Panel** appears within the Sitecore Desktop.
3. In the **Control Panel**, click **Analytics**. The **Engagement Analytics Control Panel** appears.
4. In the **Engagement Analytics Control Panel**, click **Refresh Lookups** and complete the wizard that appears.
5. In the **Engagement Analytics Control Panel**, click **Retrieve Status Information**. The **Analytics Status Information** report appears in the Sitecore Desktop.
6. In the **Analytics Status Information** report, if **Latest Success** for Requests does not indicate a recent date and time, or if **Latest Failure** for Requests indicates a recent date and time, then the system exceeded the event queue.

Note
If a reverse DNS lookup or GeoIP lookup operation times out or otherwise fails, information obtained from the reverse DNS lookup provider does not appear in Engagement Analytics reports. Check the configuration of the GeoIP lookup provider, confirm that the system has Internet access, and investigate the Sitecore log for more information.

Other options in the Engagement Analytics section of the Sitecore Desktop Control Panel enable you to Copy Configuration Data from the Analytics database to the Master database and to Update Visitor Classifications.