BusinessObjects Web Intelligence XI R1/R2: Advanced Report Design

Learner's Guide
QA320R2

Revision A, May 2006

Copyright

Patents

Business Objects owns the following U.S. patents, which may cover products that are offered and sold by Business Objects: 5,555,403, 6,247,008 B1, 6,578,027 B2, 6,490,593 and 6,289,352.

Trademarks

Business Objects, the Business Objects logo, Crystal Reports, and Crystal Enterprise are trademarks or registered trademarks of Business Objects SA or its affiliated companies in the United States and other countries. All other names mentioned herein may be trademarks of their respective owners. Product specifications and program conditions are subject to change without notice.

Copyright

C O N T E N T S

About this Course

Computer Setup

Lesson 1
Advanced Query Techniques

Lesson 2
Character and Date String Functions

Lesson 3
Using If Logic

Lesson 4
Advanced Reporting Techniques

Lesson 5
Calculation Contexts

Appendix A
Answer Key

A G E N D A

Advanced Report Design

Introductions, Course Overview..... 30 minutes

Lesson 1

Advanced Query Techniques ..... 1 hour

- ❒ Use combined queries
- ❒ Use sub-queries
- ❒ Create a query based on another query
- ❒ Lesson summary

Lesson 2

Character and Date String Functions.....1 hour 30 minutes

- ❒ Character strings
- ❒ Using the Replace() function
- ❒ Using the Right() function
- ❒ Using the SubString() function
- ❒ Using the Pos() function
- ❒ Concatenating different character strings
- ❒ Using date functions
- ❒ Lesson summary

Lesson 3

Using If Logic ..... 1 hour

- ❒ Grouping data
- ❒ Using the If() function to modify calculation behavior
- ❒ Lesson summary

Lesson 4

Advanced Reporting Techniques..... 1 hour

- ❒ Formatting breaks
- ❒ Creating custom sorts
- ❒ Displaying document data in free-standing cells
- ❒ Displaying data restricted by a filter or ranking
- ❒ Lesson summary

Lesson 5

Calculation Contexts . . . . . 1 hour 30 minutes

- ❒ Understanding calculation contexts
- ❒ Redefining calculation contexts
- ❒ Lesson summary

About this Course

This section explains the conventions used in the course and in this training guide.

Course objectives

Advanced Report Design is a classroom-based course where participants learn use advanced querying and reporting techniques to create Web Intelligence documents.The course includes presentation of concepts, demonstration of features, facilitated discussions, practice activities, and reviews.

After completing this course, you will be able to:

- Create Web Intelligence documents using advanced query techniques
- Apply character and date string functions
- Create variables using "If. . . .Then. . .Else. ." logic
- Apply advanced reporting techniques using breaks, custom sorts and functions
- Understand calculation contexts and how to redefine them

After completing this course, you will be able to:

Course audience

The target audience for this course is experienced Web Intelligence report designers.

Prerequisite education

To be successful, learners who attend this course must have attended the following offerings:

- BusinessObjects Web Intelligence XI R1/R2: Report Design

Prerequisite knowledge/experience

To be successful, learners who attend this course must have working knowledge of:

- Core concepts and functionality of Web Intelligence

Course success factors

Your learning experience will be enhanced by:

- Activities that build on the life experiences of the learner
- Discussion that connects the training to real working environments
- Learners and instructor working as a team
- Active participation by all learners

Course materials

The materials included with the course materials are:

- Name card
- Learner's Guide
  The Learner's Guide contains an agenda, learner materials, and practice activities.

  The Learner's Guide is designed to assist students who attend the classroom-based course and outlines what learners can expect to achieve by participating in this course.
- Evaluation form
  At the conclusion of this course, provide feedback on the course content, instructor, and facility through the evaluation process. Your comments will assist us to improve future courses.

Additional information for the course is provided on the resource CD or as a hard copy:

- Sample files
  The sample files on the resource CD can include required files for the course activities and/or supplemental content to the training guide.
- Overview webinar(s)
  This webinar provides details about the course and an overview of the course material.

Additional resources include:

- Online Help and User's Guide
  Retrieve information and find answers to questions using the Online Help and/or User's Guide that are included with the product.

Learning process

Learning is an interactive process between the learners and the instructor. By facilitating a cooperative environment, the instructor guides the learners through the learning framework.

What's specific to BusinessObjects XI Release 2?

The audience for this course may consist of BusinessObjects XI Release 1 and BusinessObjects XI Release 2 customers. This icon has been placed throughout the guide to identify features that are specific to BusinessObjects XI Release 2.

Introduction

Why am I here? What's in it for me?

The learners will be clear about what they are getting out of each lesson.

Objectives

How do I achieve the outcome?

The learners will assimilate new concepts and how to apply the ideas presented in the lesson. This step sets the groundwork for practice.

Practice

How do I do it?

The learners will demonstrate their knowledge as well as their hands-on skills through the activities.

Review

How did I do?

The learners will have an opportunity to review what they have learned during the lesson. Review reinforces why it is important to learn particular concepts or skills.

Summary

Where have I been and where am I going?

The summary acts as a recap of the learning objectives and as a transition to the next section.

Recommended education

This course is not part of a certification path.

Computer Setup

This section lists the hardware and software setup requirements for trying the activities on your own.

Introduction

The purpose of this setup guide is to provide the information necessary to set up your computer and ensure the necessary course files are installed if you want to recreate the learning environment.

This document provides:

- Hardware and software requirements for Business Objects training
- Guidance for learners wishing to recreate the learning environment

Computer Setup

Hardware

❑ The minimum hardware requirements are:

- P3 700MHz
- 512 MB RAM, 1 Gigabyte RAM recommended
- 5 Gigabytes available hard drive space
- CD ROM

Software

❑ The software required for this course is:

Operating system (choose one):

- Microsoft Windows 2000 Data Center Server SP4
- Microsoft Windows 2000 Advanced Server SP4
- Microsoft Windows 2000 Server SP4
- Microsoft Windows 2003 Server
- Microsoft Windows 2003 Data Center Server
- Microsoft Windows 2003 Web Server
- Microsoft Windows 2003 Enterprise Edition Server
- Microsoft Windows XP (Unsupported operating system but suitable for classroom use only, not suitable for production environment.)

System Database:

- Native-MySQL (installed with BusinessObjects Enterprise by default)
- MS SQL Server (choose one)
  - MS SQL Server 2000 SP3a or higher
  - ODBC-MS SQL Server 2000 SP3a or higher
  - ODBC-MS SQL Server 7

Web Browser:

- IE 6.0 SP2

Web Server:

- Tomcat 5.0.27 with JDK 1.4.2 (installed with BusinessObjects Enterprise by default)
- Microsoft IIS (5.0 ISAPI and CGI or 6.0 ISAPI)

Applications:

- BusinessObjects Enterprise Professional (ensure you use Processor licensing)
- Microsoft Office (version 2000 or newer)
  - Microsoft Word
  - Microsoft PowerPoint
  - Microsoft Excel
- Microsoft Access (optional)
- Adobe Acrobat Reader

❑ Install software according to the instructions in this document. Prior to installation, refer to the *Platforms.txt* document for the latest supported systems. Platforms.txt can be found on the installation CD. You can also find the most recent version of this document on the Business Objects support web site: http://support.businessobjects.com.

To install required software

1 Install one of the operating systems indicated above.

2 Install Microsoft IIS.

Note: If you are installing Microsoft Windows 2000 Microsoft IIS will be included by default. If you are installing Microsoft IIS 6.0, be sure to install ASP.NET.

3 Install one of the MS SQL Server system databases indicated above.

4 Install Microsoft Office (version 2000 or newer). If desired, install Microsoft Access.

5 Follow the installation procedures listed in the installation guide of *BusinessObjects Enterprise*.

Note: It is preferred that the classroom is set up with stand-alone installations, however, it is possible to install one server and have all students connect to that server.

Please note: during the BusinessObjects Enterprise installation, ensure that you select the following options:

- Server installation (not Client)
- New installation (not Expand or Custom)
- Select Java and .NET web installation, including Tomcat
  WARNING: If you opt to install the .NET framework, there are multiple activities in the course that will not be possible. This course REQUIRES that the Java environment be installed.

  Note: A prepackaged version of Tomcat with Apache web server is provided during the BusinessObjects Enterprise XI R2 installation.

6 Make sure that all of the computers in the training room can access the same network and can communicate with one another over the TCP/IP network.

Testing the hardware and software setup

To ensure the hardware and software are set up properly, perform these steps on your machine.

To test networking of all machines

1 From the instructor's machine, run **ipconfig** using the command prompt.

2 If it says media cable disconnected, check your hardware wiring and make sure it is plugged into the room's network at a minimum.

3 From the command-line prompt, run **ping <ip or machine name of a student machine>**.

4 If it says "request timed out" then recheck hardware wiring and make sure all machines are on the network.

5 If you can ping all machines from one single machine and all computers reply/respond then your network is fully functional.

To test Central Management Console installation

1. From the BusinessObjects XI Release 2 Program Group, click on **BusinessObjects Enterprise Java Administration LaunchPad.**
2. Click on the link for the **Central Management Console**.
3. Log on as Administrator, no password.
   Successful login indicates successful installation. Repeat steps 1-3 for .NET Admin Launchpad.

To test InfoView installation

1. From the BusinessObjects XI Release 2 Program Group, click on **BusinessObjects Enterprise .NET InfoView**.
2. Log on as Administrator, no password.
   Successful login indicates successful installation.
3. Ensure that the report samples have been installed successfully by navigating to any Public folder and opening an object.
   For example, Report Samples > General Business > Open the Product Catalog report.

Setting up for the activities

The following setup needs to be completed prior to the instructor arriving on-site. The universes below will be used during the classroom activities. The following setup instructions are NOT part of the course and must be completed prior to the course. If the items below are not completed prior to the instructor arriving on-site it WILL cause a delay in the delivery of the course and you may be charged for the additional setup time required.

The directions below are step-by-step and can be followed with little knowledge of the BusinessObjects Enterprise XI development environment.

Upon completion of these items please contact your on-site instructor to let them know that the setup is complete.

Deployment of Universes

This course uses the following universe:

- eFashion

This universe is installed by the Enterprise Setup program to the following location:

C:\Program Files\Business Objects\BusinessObjects Enterprise 11.5\Samples\en\Universes

It is also available on the course resource CD. The corresponding Microsoft Access database is also available on the course Resource CD.

In order to make the universe available to the users you must first redefine their connections to the appropriate data source (Microsoft Access database file), then export the universe to the repository.

Before doing this, you first need to ensure that a valid ODBC connection is present on the workstation.

The following instructions describe how to define a connection and export the eFashion universe.

To create an ODBC connection:

1 Click **Start > Programs > Administrative Tools > Data Sources (ODBC)**.

2 Click the **System DSN** tab.

Note: In order to deploy a universe, the connection to the data source must use a system DSN.

IF	THEN
A connection for eFashion already exists	Go to Step 3.
No connection for eFashion exists.	Go to Step 10.

Start here if an eFashion ODBC connection already exists.

3 Click **eFashion**.

4 Click **Configure**.

5 Click **Select**.

Note: The correct database may or may not be selected. If a previous version of BusinessObjects ever existed on the machine, the connection may be left over from that version. Continue through steps 5 - 8 to verify the correct database is selected. The eFashion database in BusinessObjects Enterprise 11 is different than the database in previous versions.

6 Browse to the directory you installed BusinessObjects Enterprise 11 in ...**\Business Objects\BusinessObjects Enterprise 11\Samples\En\Databases**.

7 Select **efashion.mdb**. Click **OK**.

8 Click **OK** to close the ODBC Microsoft Access Setup window.

9 Click **OK** to accept the ODBC changes.

10 Skip to step 18.

Start here if a new eFashion ODBC connection is required.

11 Click **Add**.

12 Select **Microsoft Access Driver (*.mdb)**. Click **Finish**.

13 Type **eFashion** as the **Data Source Name**.

14 Click **Select**.

15 Browse to the directory you installed BusinessObjects Enterprise XI R2 in ...**\Business Objects\BusinessObjects Enterprise 11.5\Samples\En\Databases**.

16 Select **efashion.mdb**. Click **OK**.

17 Click **OK** to close the ODBC Microsoft Access Setup window.

18 Click **OK** to accept the ODBC changes.

After creating / verifying the eFashion ODBC connection, continue to the instructions below.

To define the connection in the universe

1 Click **Start > Programs > BusinessObjects 11 > BusinessObjects Enterprise > Designer.** Log into BusinessObjects Designer. If you are using the default installation, you can use the username **Administrator** with NO password.
If the wizard opens, click **Cancel**.

2 Click **File > Open**.

3 Browse to the directory where BusinessObjects Enterprise XI R2 was installed
...**\Business Objects\BusinessObjects Enterprise 11.5\Samples\En\Universes**.

4 Click **eFashion**.

5 Click **Open**. The universe will open in Designer.

6 Click **File** > **Parameters**.

7 On the Definition tab, click on **New** to add a new connection.

Note: The connection must be a "secured" connection prior to publishing to the repository.

8 The New Connection Wizard appears. Click on **Next**.

9 Double-click **Microsoft**.

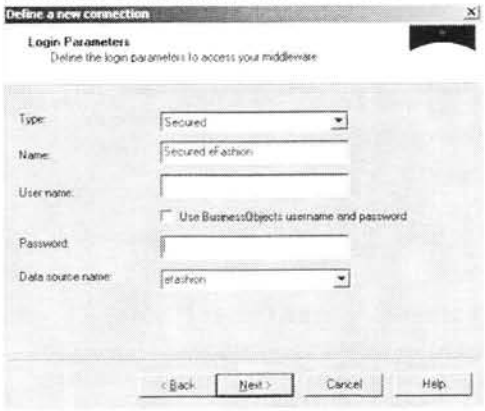
10 Double-click **MS Access 2002.** (Or the appropriate MS Access version)

11 Click **ODBC Drivers**. Click **Next**.

12 Verify the **Type**: is **Secured**.

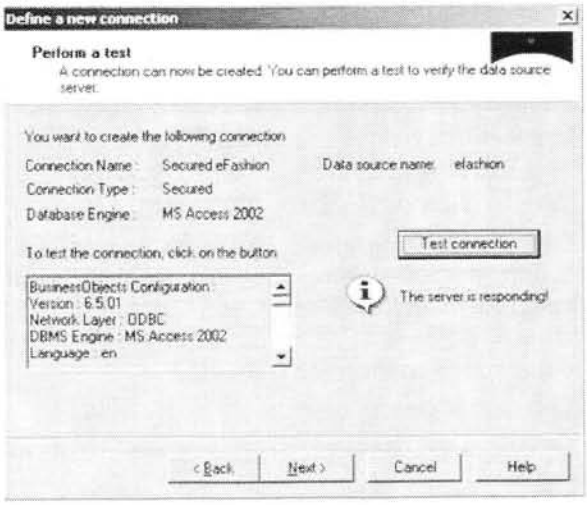
13 Type **Secured eFashion** as the **Name**.

14 Select **eFashion** as the **Data source name**.
Your new connection window should look like this:



15 Click **Next**.

16 Click **Test Connection**. Verify that the test was successful.
If you do not see the message The server is responding! Click the < Back button and resolve the issue.

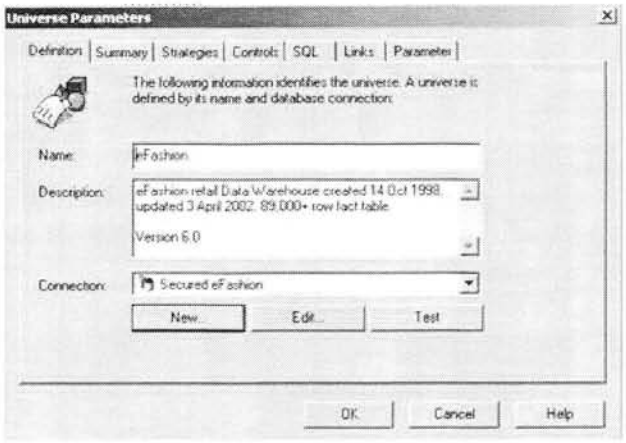


17 Click **Next**.

18 Click **Next**.

19 Click **Finish**.

The **secured eFashion** connection is now shown.



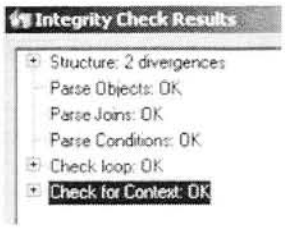
20 Click **OK**.

21 Click **Tools > Check Integrity**.

22 Check the check box for **Check All**.

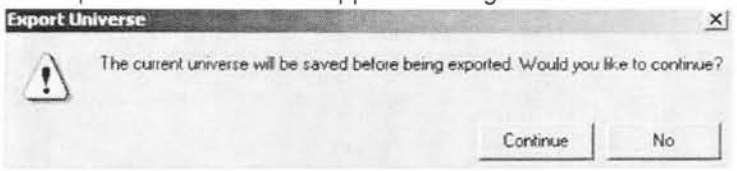
23 Click **OK**. The integrity check runs.

24 Verify that the **Check Integrity Results** match exactly as shown below.



25 Click **OK** to close the window.

26 Click **File > Export**.
The export universe window appears asking for confirmation of saving.

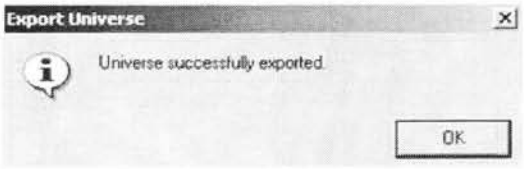


27 Click **Continue**.

28 Select the **Folder**.

29 Select the **Groups** (leave default if you are not sure).

30 Click **OK**.
The Universe successfully exported message appears.



31 Click **OK**.

32 Close BusinessObjects Designer.

Note: As stated previously, the eFashion sample universe and database are located in this folder:

C:\Program Files\Business Objects\BusinessObjects Enterprise 11.5\Samples\en\

Testing the hardware and software setup

After exporting the eFashion universe, follow the steps below to verify that all files are in place for the training class.

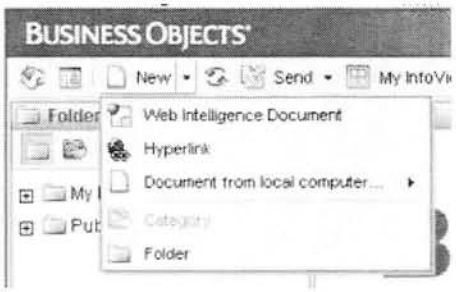
Verify users can log into the JSP version of BusinessObjects Enterprise XI R2

1 Click Start > Programs > BusinessObjects XI Release 2> BusinessObjects Enterprise > BusinessObjects Enterprise Java InfoView.

2 Log in as a valid user. If running only the samples, log in as **Administrator**. Administrator does not have a password.

Verify eFashion universe is available.

3 Click **Web Intelligence Document** from the **New** menu.

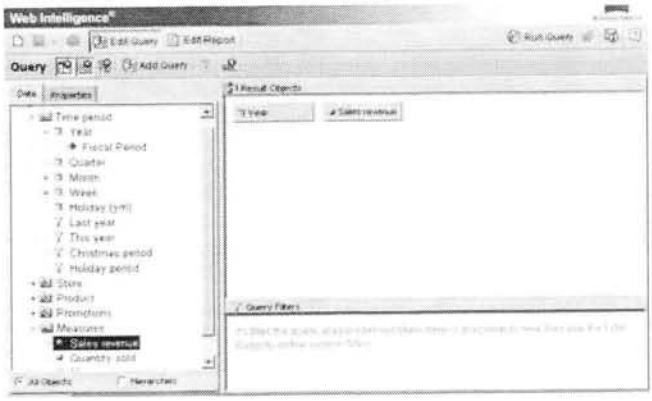


4 Verify that **eFashion** is listed as an available universe.

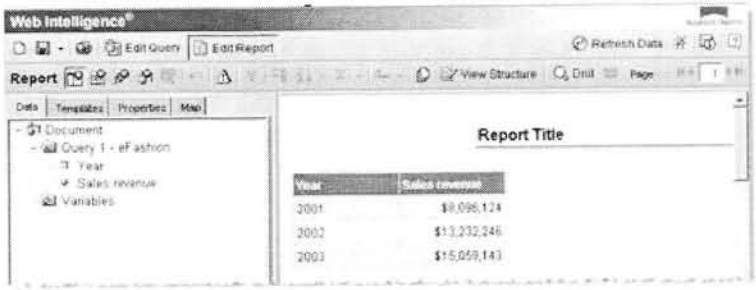
5 Click **eFashion** to create a new Web Intelligence document.
If the security warning appears, click **Yes** to trust the applet.

6 Double-click **Year**.

7 Double-click **Sales Revenue** (in the Measures class).



8 Click **Run Query**.
The results should match the image below.

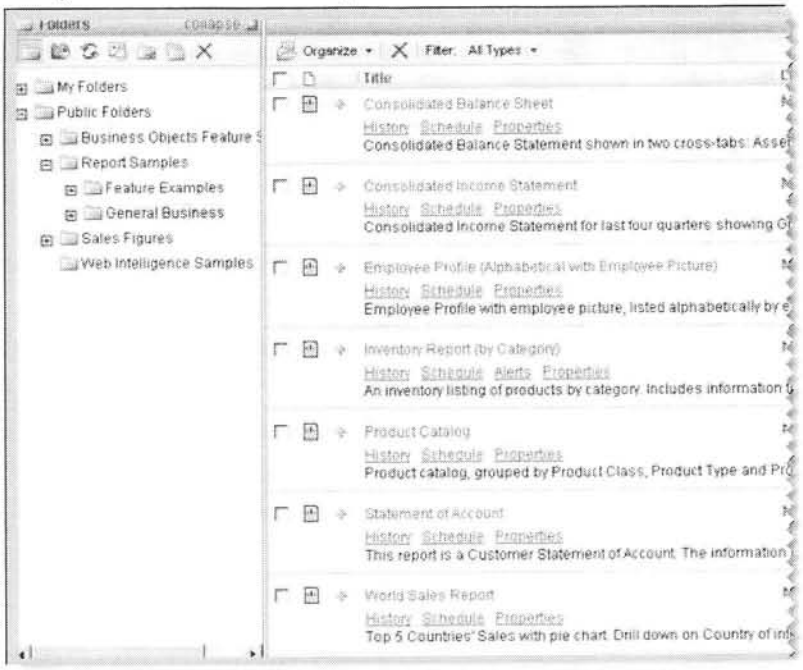


9 Close the document without saving (click X in top right corner).

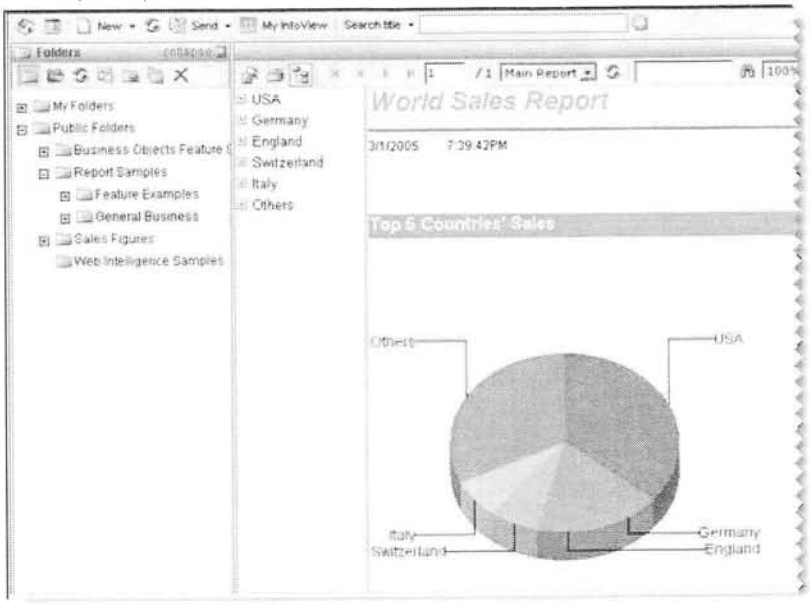
Verify sample Crystal Report document is available.

10 Click + to expand **Report Samples**.

11 Click **General Business**.
Verify that the World Sales Report is listed.



12 Click **World Sales Report** to open the report.
The report opens.



13 Close the report.

14 Log out of BusinessObjects Enterprise.

The setup is now complete.

Getting Help

If you encounter difficulties installing the software when recreating the learning environment, refer to the Installation document found on the product CD or contact Business Objects Customer Support. For a current contact list visit http://support.businessobjects.com.

Lesson 1
Advanced Query Techniques

This lesson demonstrates how to use advanced query techniques.

In this lesson you will learn how to:

- Use combined queries
- Use sub-queries
- Create a query based on another query

Duration: 1 hour

Use combined queries

Introduction

In most instances, you only need to make one Web Intelligence query to populate the data provider with the information you want to report on. However, there are certain instances where you will need to make two or more queries to retrieve the results you require. As a Web Intelligence user, there are two techniques available that enable you to do this. They are:

- Combined queries
  The combined queries technique enables you to create two queries and merge the results of both into a single data provider on a selective basis. This can be done by using the Union, Intersect, or Minus functions.

  Note: In this unit, you will see that combining queries in a single data provider is different from synchronizing multiple data providers using merged dimensions, which was presented in the *BusinessObjects Web Intelligence XI R1/R2 Report Design* course.

- Sub-queries
  The subquery technique enables you to specify the output of a query as the operand value(s) for a query filter of another query.

This unit describes how to use combined queries.

After completing this unit, you will be able to:

- Describe the Union, Intersection and Minus functions used to combine queries
- Discuss reasons and advantages to using combined queries instead of applying complex filters
- Create a combined query
- Remove a combined filter
- List important facts to remember when using combined queries

About combined query functions

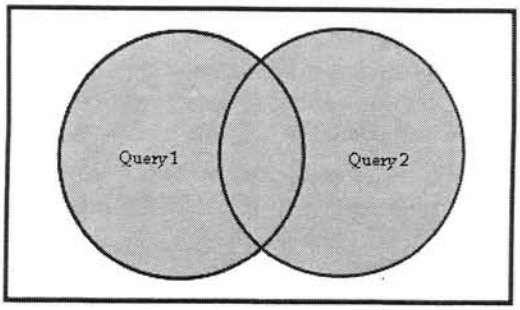
Adding single and complex query filters to a query allows you to restrict the amount of information returned by the query. This standard technique allows you to combine multiple filters in a single query, and these combinations can be designed to make queries very specific and limiting.

However, this technique only works with a single query. There may be situations when you want to combine the results of two queries into a single block.

To accomplish this, you must build a *combined query.* All combined queries are built in the Edit Query view of the Web Intelligence Java Report Panel, and can only be built using a single universe. You can create up to eight queries from the same universe; however, each query needs to have the same number of objects as well as the same data types.

There are three methods of combining queries:

- Union: combine results which appear in Query 1 OR Query 2.

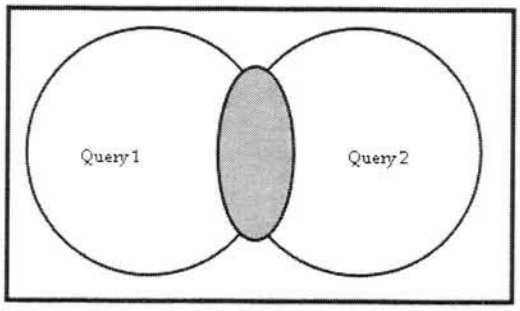


  You use a union query to combine the data from two objects in a single column in a table. Union queries are especially useful for working with incompatible objects.

  For example, if you built a query with two incompatible objects, Web Intelligence would run separate SQL statements for each object and then return the data in different blocks (tables). A union query forces Web Intelligence to return the data from both objects together in one column.

  Note: Union is the default operator for combined queries.

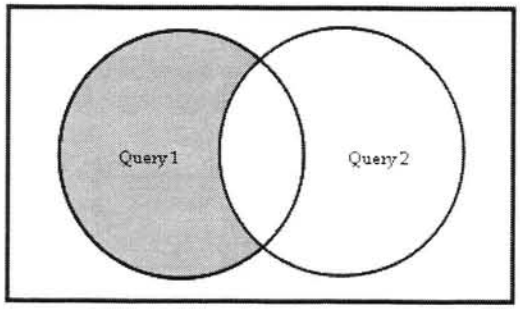
- Intersection: combine results which appear in Query 1 AND Query 2.



  You use an intersection query to obtain data common to two sets of results. Like a union query, Web Intelligence considers each select statement separately and combines their results in the end.

  Unlike a union query, the intersection query only returns those values that are in both queries. In this sense, it works much the same as using the AND operator when creating a regular query with multiple query filters.

- Minus: combine results which include everything in Query 1 except for what is also true in Query 2.



You use a minus query to exclude the results of one query from the main query result (Query 1). For example, a minus query could be used to find out which customers bought product A but not product B.

You could not obtain this data with standard query filters since the result sets need to be obtained separately before being combined. Like a union query, the minus query considers each query separately and combines their results in the end.

When you build minus queries, you must pay attention to the order of the queries, since the results of Query 2 will always be subtracted from the results of Query 1.

Understanding when to use a combined query

The following is a typical scenario where you might choose to use a combined query:

You need to find the dates on which your customers either made reservations or paid their invoices. The query requires two incompatible objects: Invoice Date and Reservation Date. Since the objects are incompatible, if you include them in a standard query, Web Intelligence returns two blocks of data and does not synchronize the values in a single block. However, by building a combined query using the Union function, with Invoice Date in the first query and Reservation Date in the other, the data appears in a single column in the block.

The Union function is much the same as using the OR operator with a complex query filter: it returns dates on which either invoices were paid OR reservations were made.

However, the column of data retrieved by the combined query displays a header that reflects the date object from the first query only. In other words, the query returns a column entitled Invoice Date, but the column actually contains both Invoice Dates and Reservation Dates. This is important to understand, as it may require you to reformat the header.

Advantages of using combined queries

There are a number of reasons why you may want to use combined queries instead of applying multiple query filters against a single query:

- To make the construction of the query easier
- When it is not possible to set the required query filters using Boolean logic

The downside of using the combined query technique is that because you are actually creating multiple queries, processing against the database may take longer.

Note: A decision on whether to use query filters or a combined query to retrieve the data you need often depends on how the data is structured in your database.

Comparing query filters and combined queries

As an e-Fashion Regional Product Manager, you want to compare accessories sold in California and those sold in Colorado.

To do this, you can use one of three techniques:

- You can create two separate queries and display the data in two different blocks, either in the same report or in separate reports in the document.
- You can create a single query and apply a complex filter.
- You can combine two different queries into a single query.

The technique you choose depends largely on the data that you are working with. However, using the combined query technique allows you the flexibility of comparing the data in a single block, and in different ways, using the union, intersection or minus function, depending on how you want to display the data.

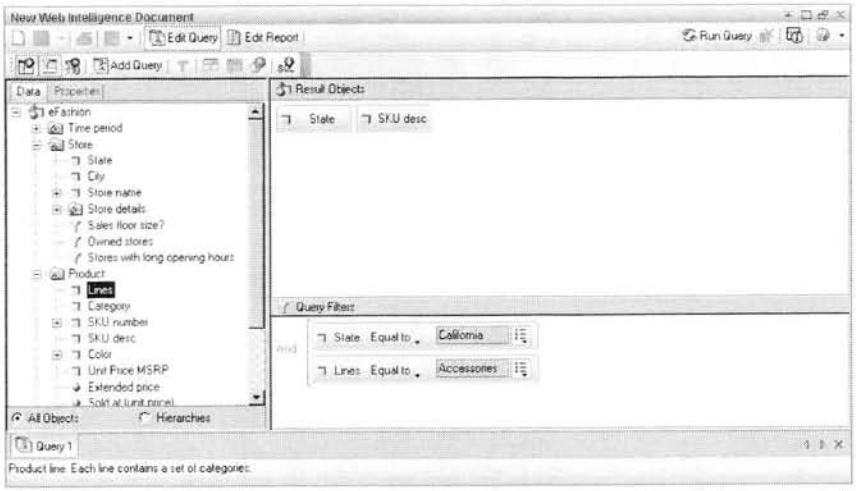
To display the results by building two queries

In this example, you build one query to show accessories sold in California, and then a second query that shows accessories sold in Colorado. You display the data in separate blocks in a single report.

1 Create a new Web Intelligence document using the **eFashion** universe.

2 In the Java Report Panel, build a query by dragging or double-clicking the **State** and **SKU desc** objects to move them into the Result Objects pane.

3 Apply a query filter so that only data concerning **California** and the **Accessories** product line is retrieved.



4 Click **Run Query**.

5 In the Edit Report view, apply a **Break** on the **State** column and a **Count** to the **Sku desc** column in the resulting block.

	Strawberry Patterned Viscose Scarf
	Striped Leggings
	Suede Belt with Heart Shaped Buckle
	Suede Cloth Belt
	Sunflower Patterned Viscose Scarf
	Thick Silver Bangle
	Thinsulate Knit Gloves
	Tie-die Patterned Viscose Scarf
	Tortoiseshell Brooch
	Tulip Patterned Viscose Scarf
	Two Color Double Sided Velvet Scarf
	Velvet Purse
	Velvet Shoulder Bag
	Zipper Vest
California	**75**

You will find that there are 75 separate rows of product data in the table.

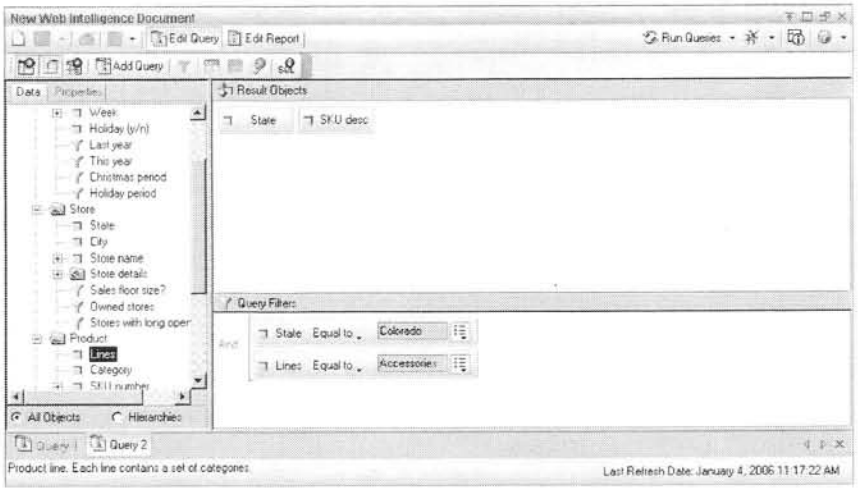
Note: The above image does not show all of the rows in the table.

6 Click **Edit Query** to return to Edit Query view and add a second query to the same document.

7 In Edit Query view, click the **Add Query** button.
The Universe dialog box appears, which allows you to select the universe you want to use for the new query.

8 Select the **eFashion** universe.

9 Build the same query, except change the query filter to retrieve data for **Colorado**, instead of California.



10 Click the drop-down arrow next to the **Run Queries** button and click **Query 2**.

11 Select the **Include the result objects in the document without generating a table** option.

12 In Edit Report view, at the bottom of the **Data** tab, click the **Arranged by: Alphabetical order** drop-down button.

13 Select the **Query** option from the drop-down menu that appears.
The objects listed in the Data tab now appear organized by query, instead of in alphabetical order.

14 Click + next to the **Query2 - eFashion** folder to see the objects used in your second query.
The Query2 - eFashion folder expands.

15 While holding down the **Ctrl** key, click both the **State** and **SKU desc** objects in the **Query2 - eFashion** folder and drag them over to the empty space next to the existing table.
The new table displays the results for Colorado.

16 In the new table, apply a **Break** on the **State** column and a **Count** to the **Sku desc** column in the second block.

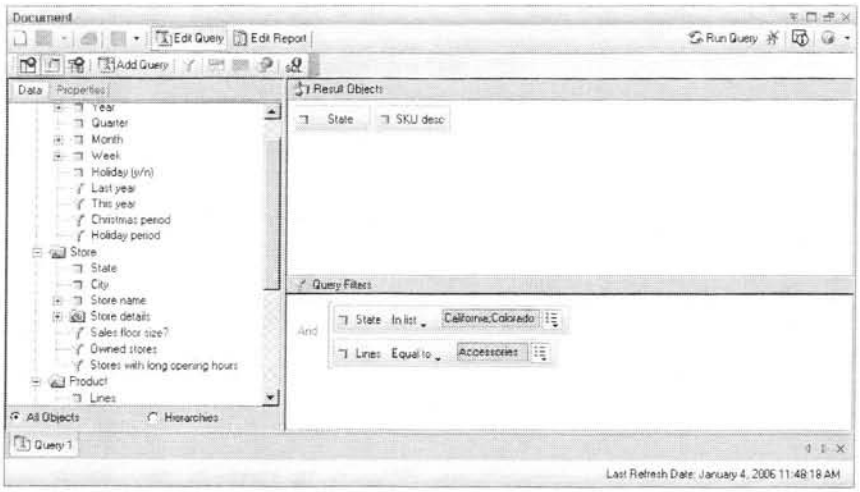
	Sunflower Patterned Viscose Scarf
	Thick Silver Bangle
	Thinsulate Knit Gloves
	Tie-die Patterned Viscose Scarf
	Tortoiseshell Brooch
	Tulip Patterned Viscose Scarf
	Two Color Double Sided Velvet Scarf
	Velvet Shoulder Bag
	Zipper Vest
Colorado	**57**

Note: The above image does not show all of the rows in the table.

Notice that the new table shows that you have sold 57 accessories in Colorado.

To display the results by applying a complex query filter to a single query

You can get the same result by building a single query, and applying a complex query filter that specifies both California and Colorado in the query filter definition, as shown here:



In the next section, you will see how using combined queries allows you to analyze the commonalities and differences in the accessories sold across the two states.

Using the combined query technique

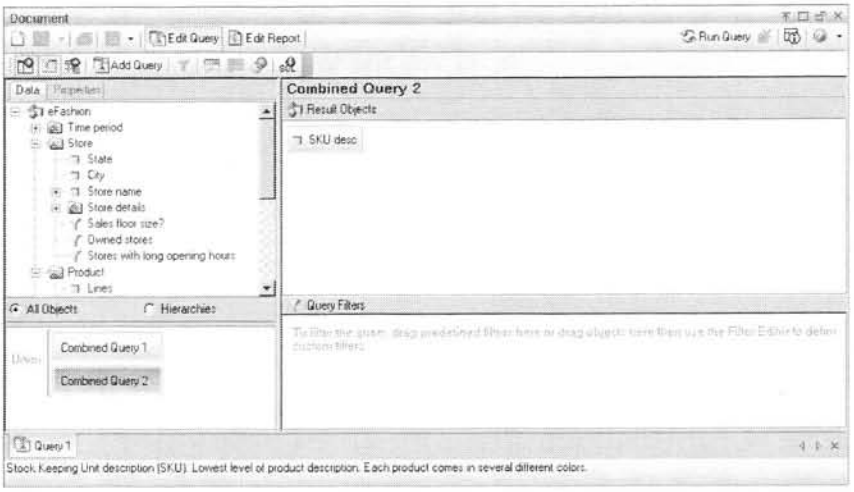
The combined queries technique combines the results of one query with the results of another query into a single query, or data provider. The manner that the data is combined depends on the function you choose: Union, Minus or Intersection.

To combine queries

1 In the Java Report Panel, click the **New Document** button to create a new document using the same universe.

2 In Edit Query view, drag or double-click the **SKU desc** object to the Result Objects pane.

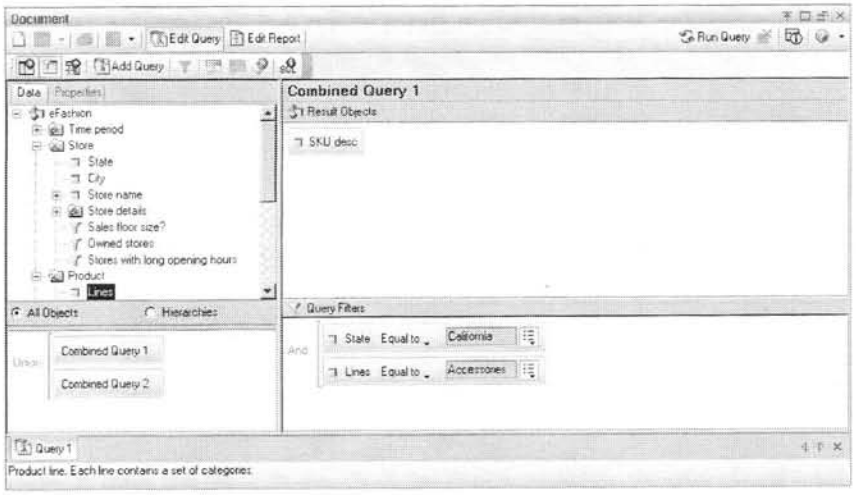
3 Click the **Combined Queries** button.
A new pane opens in the Edit Query view, just below the Classes and Objects pane.



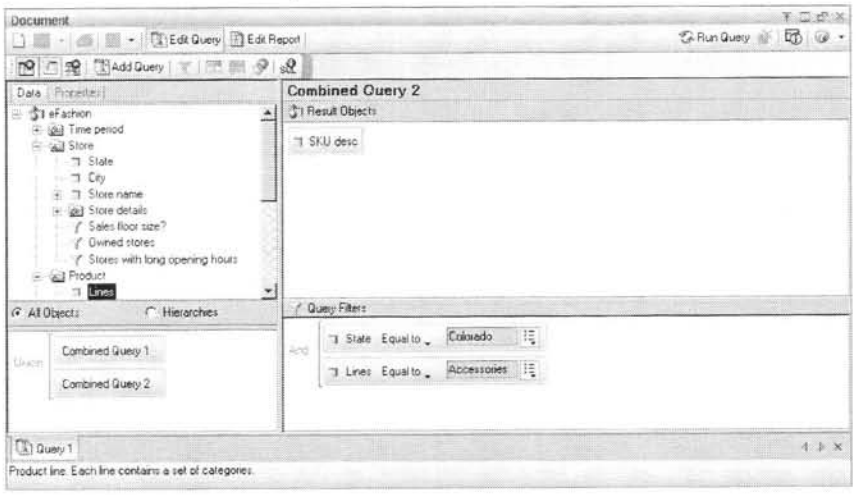
This pane allows you to define how you want to combine the queries that will be included in this single data provider. Note that the Combined Query 2 label in this pane is highlighted and the Result Objects pane is currently showing the query definition for Combined Query 2.

4 In the Combined Query pane, click the **Combined Query 1** label to display its query definition.
The Combined Query 1 definition appears in the Result Objects pane.

5 Insert the query filters illustrated in this example in the Query Filters pane of Combined Query 1.

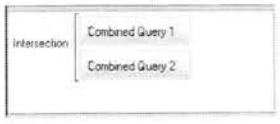


6 Click the **Combined Query 2** label, and then apply the query filter as illustrated in this example:

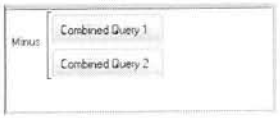


By default, the system applies a Union function between Query 1 and Query 2, as shown in the Combined Queries pane.

You can change the function double-clicking Union in the Combined Queries pane. This toggles the function to Intersection.



Double-clicking the function once again toggles it to Minus.



In this instance, you want to use the default function, the Union function.

7 Click **Run Query**.

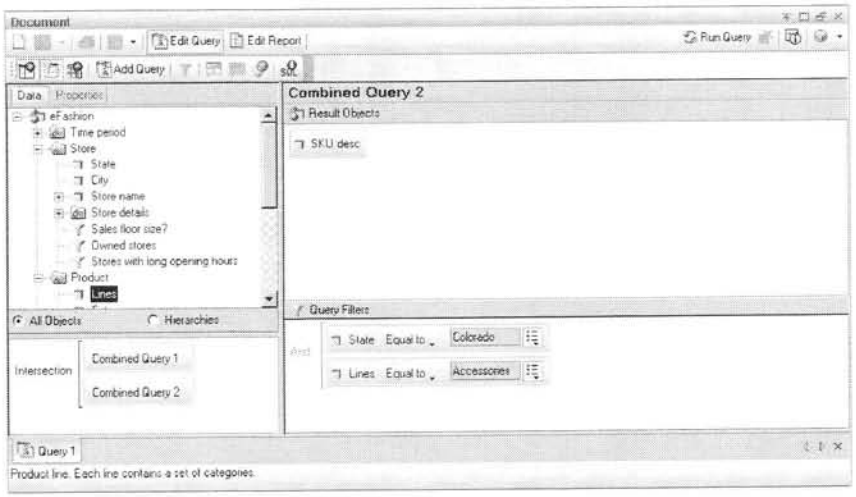
8 Apply a count to the single column of data.

Suede Cloth Belt
Sunflower Patterned Viscose Scarf
Thick Silver Bangle
Thinsulate Knit Gloves
Tie-die Patterned Viscose Scarf
Tortoiseshell Brooch
Tulip Patterned Viscose Scarf
Two Color Double Sided Velvet Scarf
Velvet Purse
Velvet Shoulder Bag
Zipper Vest
75

Notice there are 75 Accessories in your list. These are the ones that were sold in either California or Colorado.

Note: The above image does not show all of the rows in the table.

9 Edit the query so that it uses the **Intersection** function to combine the two query definitions.



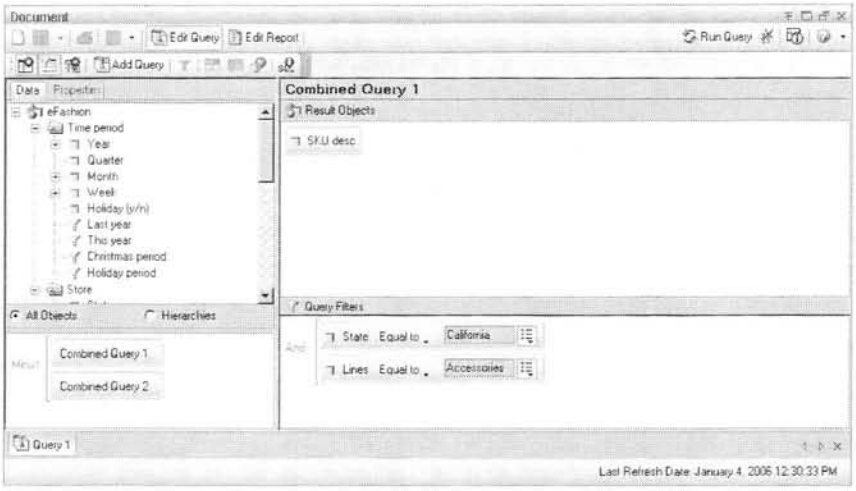
10 Click **Run Query**.

Spotty Leggings
Square Viscose Sample
Striped Leggings
Suede Belt with Heart Shaped Buckle
Sunflower Patterned Viscose Scarf
Thick Silver Bangle
Thinsulate Knit Gloves
Tie-die Patterned Viscose Scarf
Tortoiseshell Brooch
Tulip Patterned Viscose Scarf
Two Color Double Sided Velvet Scarf
Velvet Shoulder Bag
Zipper Vest
57

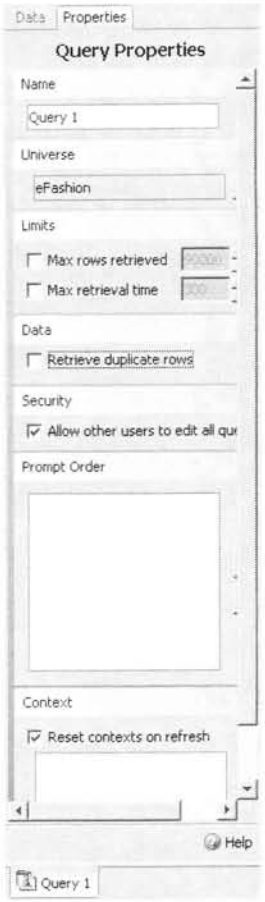
Notice there are 57 accessories in the new list. These are the ones that were sold in both California and Colorado.

Note: The above image does not show all of the rows in the table.

11 Edit the query so that it uses the **Minus** function as its combined query technique.



12 In the Query Properties tab, clear the Retrieve duplicate rows box.



13 Click **Run Query**.
Notice there are 18 accessories in the new list. These are the ones that were sold in California, excluding those that were *not* sold in Colorado.

Chain and Pearl Necklace
Diamond Star Earrings
Goldset Pearl Necklace
Hair Clip
Jacquard Scarf
Large Rose Patterned Viscose Scarf
Large Silver Bracelet
Pastel Colored Viscose Scarf
Rigid Twisted Knot Bracelet
Shell Headband
Silver Hoop Earrings
Snake Loop Necklace
Strawberry Patterned Viscose Scarf
Suede Cloth Belt
Velvet Purse
18

Note: The above image does not show all of the rows in the table.

14 Save this document to your Favorites folder as **Combined Query**.

To remove the combined query

1. In Edit Query view, right-click either the Combined Query 1 label or the Combined Query 2 label.
2. Select Remove from the drop-down menu.
   The query definition you highlighted is deleted. Whichever query definition you did not right-click will remain in the Result Objects and Query Filters pane.

Important facts about combined queries

Below are some important facts to keep in mind when you use combined queries:

- Queries that you combine must contain the same number of objects in order to return the same number of columns of data.
- When you build a combined query to return data from more than one object in a column, for example, using the union operator, you must use objects of the same type (character, date or number).
- Only the objects placed in the first query are displayed in the block after the query is run. Any object used in a combined query does not display in its own column, but instead, the values are returned in the same column as the object from the first query.
- Web Intelligence's default behavior when combining queries is to perform a Union (not a UNION ALL), and to return all values less the duplicates.
- You can include up to eight queries in a combined query.

Practice

Activity: Combined queries

Objectives

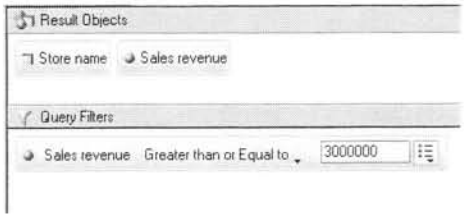
Create a Web Intelligence document by combing queries, using the Union, Intersection and Minus functions.

Instructions

You are interested in reporting on the relationship between the eFashion stores that earned at least $3,000,000 in sales revenue and those stores whose margin was at least $1,300,000.

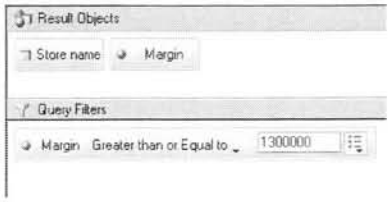
Assume that you already have two reports built using the following queries:

Report 1:l



Store name	Sales revenue
e-Fashion Chicago 33rd	$3,022,658
e-Fashion Houston Leighto	$3,144,774
e-Fashion Los Angeles	$4,220,929
e-Fashion New York Magno	$4,621,854
e-Fashion San Francisco	$3,258,641
5	

Report 2:l



Store name	Margin
e-Fashion Los Angeles	$1,668,385
e-Fashion New York Magnolia	$1,870,868
e-Fashion San Francisco	$1,304,515
3	

Using the combined query technique, create the following reports:

1 Create a table showing all Store names that have had both Sales Revenue of $3,000,000 or higher **and** a margin of $1,300,000 or higher.
   - Which operator will you use to combine the queries?

2 Once you run the combined query, apply a **Count** to the **Store name** column.
The table should appear like this:

Store name
e-Fashion Los Angeles
e-Fashion New York Magnolia
e-Fashion San Francisco
3

3 Edit the query and update the data in the table so that it shows all Store names that have had Sales revenue of at least $3,000,000 but **not** a margin of $1,300,000 or more.

- Which operator will you use to combine the queries?

4 Once you run the combined query, check that the **Count** is still applied to the **Store name** column.

The table should appear like this:

Store name
e-Fashion Chicago 33rd
e-Fashion Houston Leighton
2

5 Edit the query to display all store names that have had either Sales revenue of at least $3,000,000 **or** a margin of at least $1,300,000.

- Which operator will you use to combine the queries?

6 Once you run the combined query, check that the **Count** is still applied to the **Store name** column.
The table should appear like this:

Store name
e-Fashion Chicago 33rd
e-Fashion Houston Leighton
e-Fashion Los Angeles
e-Fashion New York Magnolia
e-Fashion San Francisco
5

7 Save the document in your Favorites folder as **Act_Combinedqueries**.

Use sub-queries

Introduction

A sub-query, as its name suggests, is a query within a query. It contains an inner query, which returns a set of data that is used as the basis for a second, outer (or main) query.

After completing this unit, you will be able to:

- Describe sub-queries
- Create a sub-query

Understanding sub-queries

Sub-queries are used in cases where the results of the main query are dependent upon the results of the inner query. This means that the inner query must be processed first so that the result set can be passed on the main query.

Like combined queries, sub-queries are always built in the Edit Query view of the Java Report Panel.

You construct a sub-query by placing a query filter on one of the objects in the main query and then using the operand for that query filter to launch the sub-query. The operator you include in the query filter determines the relationship between the data sets returned by the inner and outer queries.

Typically you use sub-queries when:

- The value of the operand is unknown.
- The query filter for the report involves a value that will change over time.

Creating a sub-query

Using this scenario, you will explore how and when to use sub-queries.

You have been asked by the Manager of the Chicago 33rd store of the eFashion Group to produce a report that lists stores and their revenue whenever those stores' revenue are higher than Chicago 33rd.

To produce the requested revenue report, you need to resolve one issue regarding the query filter. The query filter for the report involves a value that is not known prior to a query being made, and it will change over time. Therefore, you cannot enter a hard-coded figure for the sales revenue for Chicago 33rd because the user would never be able to refresh the report and get accurate results.

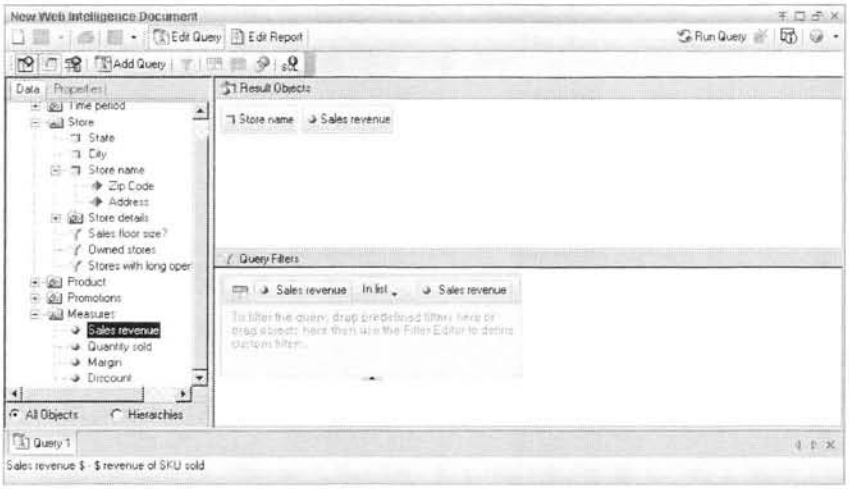
You can resolve this using a sub-query.

To apply a sub-query as part of a query

1 Create a new document using the eFashion universe.

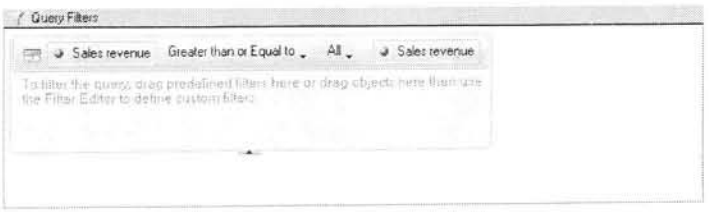
2 In the Java Report Panel, drag or double-click the **Store name** and **Sales revenue** objects to move them to the Result Objects pane.

3 Click the **Add a subquery** button.
The Edit Query view appears like this:



By default, the Sales revenue object appears in the sub-query definition.

4 Replace the default In List operand with the **Greater than or Equal to** operand.
The Query Filters pane appears like this:

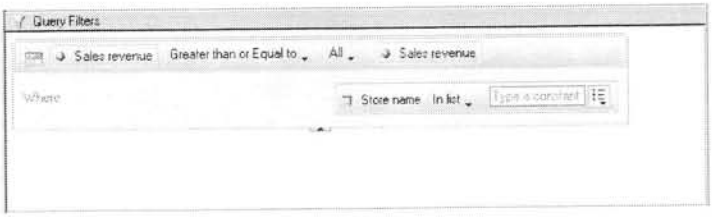


There are two sub-query operand types: All and Any.

- The All option means that the query filter object will be compared to all values resulting from the sub-query.
- The Any option means that the system will only look for one instance of a value from the sub-query that meets the query filter. If it finds one, it then ceases the comparison with the remainder of the sub-query output. Obviously, the difference between the two only matters if there is more than one value output by the sub-query. In this instance, that is not the case.

5 Leave the sub-query option to the default value, **All**.

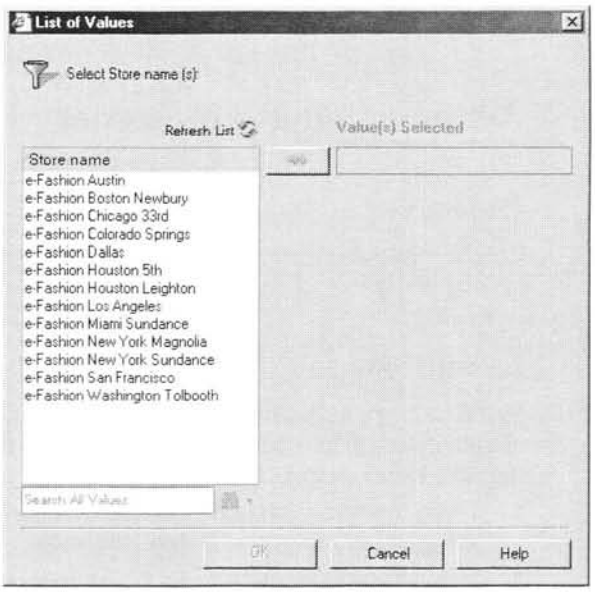
6 To define the sub-query, drag the **Store name** object into the sub-query zone just below the query filter definition.
The Query Filters pane appears like this:



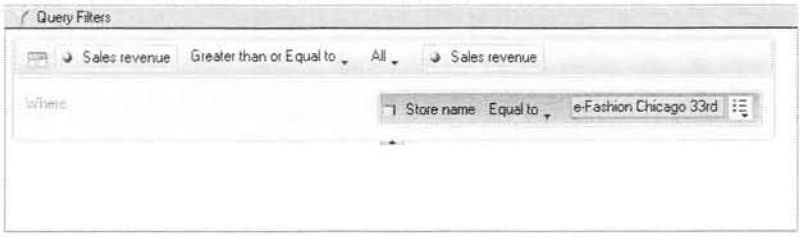
7 Replace the default In List operand with **Equal to**.

8 Click the drop-down list arrow next to the **Type a constant** field.

9 From the drop-down menu that appears, select **Value(s) from list**.
The List of Values dialog box appears:



10 From the List of Values dialog box, double-click the **eFashion Chicago 33rd** store name and click **OK**.
The Query Filters pane appears like this:



11 Click **Run Query** to process the query.
The table block displays.

Store name	Sales revenue
e-Fashion Chicago 33rd	$3,022,658
e-Fashion Houston Leighton	$3,144,774
e-Fashion Los Angeles	$4,220,929
e-Fashion New York Magnolia	$4,621,854
e-Fashion San Francisco	$3,258,641

12 Save the document as **Doc_Revenuereport**.

Note: Sub-queries involve complicated SQL and can take a long time to process. Almost all sub-queries will take longer to process than a standard query.

Practice

Activity: Using sub-queries

Objectives

Create a sub-query using the results of one query as the starting point for a second query.

Instructions

You have been asked by the Manager of the Colorado Springs store of the eFashion Group to produce a report that lists stores and their revenue where they are higher than Colorado Springs.

1 Using the eFashion universe, create a new document and build a query using **Store name** and **Sales Revenue**.

2 Using the sub-query technique, return those stores with a higher revenue than the Colorado Springs store.

3 You also need to display a title.

4 Save the report as **Act_Subqueries**.

5 The report should look like this:

Stores with a Revenue Higher than Colorado Springs

Store name	Sales revenue
e-Fashion Austin	$2,699,673
e-Fashion Chicago 33rd	$3,022,658
e-Fashion Houston 5th	$2,303,183
e-Fashion Houston Leighton	$3,144,774
e-Fashion Los Angeles	$4,220,929
e-Fashion New York Magnolia	$4,621,854
e-Fashion New York Sundance	$2,960,367
e-Fashion San Francisco	$3,258,641
e-Fashion Washington Tolbooth	$2,961,950

Create a query based on another query

Introduction

Web Intelligence enables you to base one query on the structure of another. Creating a query on a query can save time with complex reports and can ensure that different queries are defined in exactly the same way.

After completing this unit, you will be able to:

- Duplicate a query in a document to build another query

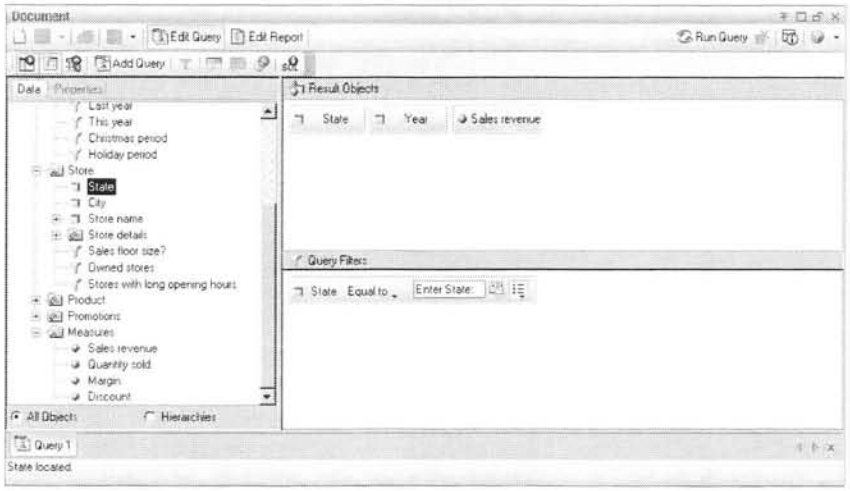
Basing a query on the structure of another

To duplicate a query in order to build another

1 Using the eFashion universe, create a new document in the Java Report Panel.

2 Drag or double-click the **State, Year** and **Sales Revenue** objects to move them into the Result Objects pane.

3 Apply a prompted query filter so that you must select a state every time the document is opened or refreshed.
The Edit Query view appears like this:



4 Run the query and select **New York**.
Your report should look like this example:

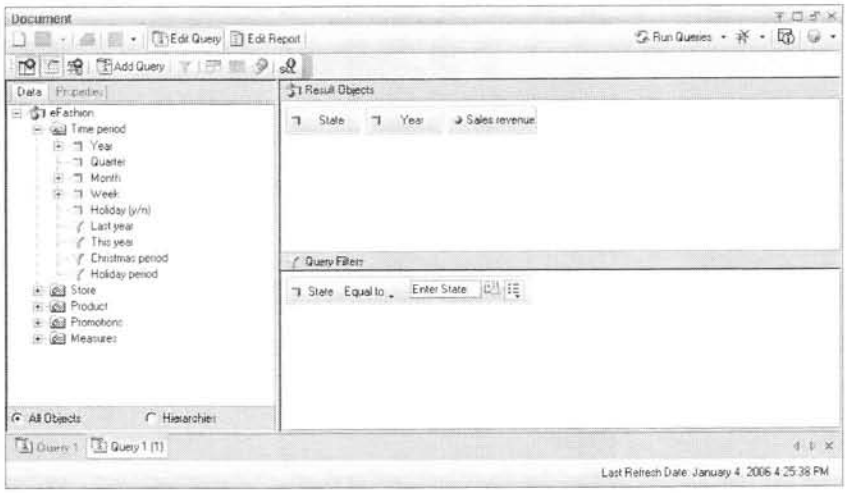
State	Year	Sales revenue
New York	2001	$1,667,696
New York	2002	$2,763,503
New York	2003	$3,151,022

Now you will set up your second report block based on the query definition used to create the first block.

5 Click **Edit Query**.

6 In the Edit Query view, right-click the **Query 1** tab at the bottom of the Data tab.

7 From the drop-down menu, select **Duplicate Query**.
A second query appears in the Edit Query view, and its tab is labelled Query 1 (1), which indicates that it is an exact copy of Query 1.



You can now edit the second query however you wish.

8 Replace the State object with the **Store name** object.

9 Click the drop-down arrow next to the **Run Queries** button.

10 Select **Query 1 (1)**.

11 Select **New York**.

12 Select the **Insert a table in the current report** option and click **OK**.
The new table appears below the previous one.

13 Using drag and drop, move the new table so that it is positioned beside the existing one.

Year	Sales revenue
2001	$1,667,696
2002	$2,763,503
2003	$3,151,022

Store name	Year	Sales revenue
e-Fashion New York 5th	2001	$644,635
e-Fashion New York 5th	2002	$1,076,144
e-Fashion New York 5th	2003	$1,239,587
e-Fashion New York Magnolia	2001	$1,023,061
e-Fashion New York Magnolia	2002	$1,687,359
e-Fashion New York Magnolia	2003	$1,911,434

Lesson summary

Review

Quiz: Advanced Query Techniques

1 What are the three types of query techniques used in this lesson?

2 If you were to use the UNION operator to combine queries, what would be the result?

3 If you were to use the INTERSECTION operator to combine queries, what would be the result?

4 If you were to use the MINUS operator to combine queries, what would be the result?

5 Give an example of why a query filter won't return data when a combined query using the MINUS does?

6 Can you do a sub-query and return exactly the same results as a combined query?

7 Why would you choose to do a combined query rather than a sub-query?

Summary

After completing this lesson, you are now able to:

- Describe the Union, Intersection and Minus functions used to combine queries
- Discuss reasons and advantages to using combined queries instead of applying complex filters
- Create a combined query
- Remove a combined filter
- List important facts to remember when using combined queries
- Describe sub-queries
- Create a sub-query
- Duplicate a query in a document to build another query

Lesson 2

Character and Date String Functions

The ability to create formulas and define them as variables in Web Intelligence offers the report designer a very powerful tool. Variables act just like dimension or measure objects. Once you have created a variable, you can use it throughout the document to display data that you cannot retrieve by using the existing objects in the universe.

In this lesson, you learn about some of the formula functions available for manipulating character and date variables. While the syntax may take some time to comprehend, it is well worth the effort as variables can provide you with more flexibility in reporting.

Note: For more information about creating formulas and defining them as variables, refer to the *BusinessObjects Web Intelligence XI R2/R2 Report Design* course.

In this lesson, you will learn about:

- Character strings
- Using the Replace() function
- Using the Right() function
- Using the SubString() function
- Using the Pos() function
- Concatenating different character strings
- Using date functions

Duration: 1 hour 30 minutes

Character strings

Introduction

A character string is a series of characters that form either a piece of text or the individual values of a character-type variable. They are always categorized as dimension objects, but the reverse is not true. A dimension object can be of character, date, or numeric type.

Using Web Intelligence character-string functions, you can replace, modify, or remove either all or part of a character string.

After completing this unit, you will be able to:

- Describe the character-string functions presented in this lesson

About character-string functions

The character-string functions presented in this lesson are:

- **Replace** - used to replace a specified string with another string
- **Right** - used to extract a given number of characters from the right
- **Left** - used to extract a given number of characters from the left
- **SubStr** - used to extract a string of variable length and position
- **Length** - used to identify the length of a string
- **Pos** - used to identify the position of a character in a string

Replace, Right, and Left are only useful when you want to manipulate a constant string or a constant number of characters. SubStr is a more powerful function. You will use the SubStr function to extract strings where the length and position are not consistent.

Using functions in formulas and variables

You use the functions presented in this lesson when you are creating a formula. You can create a formula either by typing it in the Formula bar (in Edit Report view), or in the Formula Editor. Either method calculates the data and displays the resulting values in the column that you have highlighted in the block.

In this case, the formula will be used only to display the calculated data in the highlighted column or row.

If you want to use the formula repeatedly in different blocks or different reports in the document, or if you want to identify the formula as though it were an object in the document, you can define it as a variable.

Using the Replace() function

Introduction

The Replace() function replaces a specified string with another string.

The syntax for the replace function is:

```
Replace([object] ; "old string" ; "new string")
```

String refers to the string from which the substring is to be extracted. This is normally the name of an object (for example, [Quarter]).

Old string defines the character string to be replaced.

New string refers to the character string that replaces the old string.

After completing this unit, you will be able to:

- Use the Replace() function to replace "Q" with the word "Quarter"

Replace a substring

Example

You have a requirement to report on quarters using the word Quarter instead of Q in the reports. Use the Replace() function to do this.

To replace a substring

1. Using the eFashion universe, create a new Web Intelligence document.
2. Build a query in the Java Report Panel using the **Year**, **Quarter**, and **Sales Revenue** objects.
3. Click **Run Query**.
4. In Edit Report view, right-click the **Year** column and select the **Set as Section** option from the drop-down menu.
5. Format the report so it looks like this example:

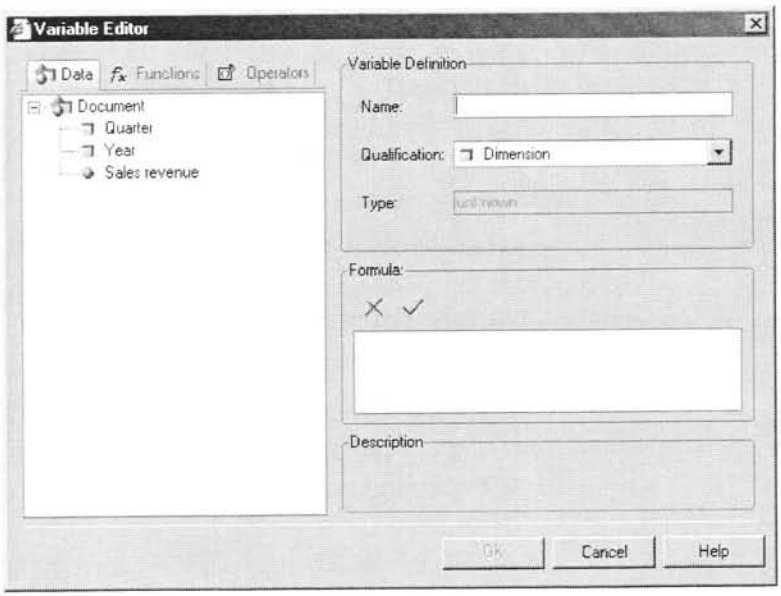
2001

Quarter	Sales revenue
Q1	$2,660,700
Q2	$2,279,003
Q3	$1,367,841
Q4	$1,788,580

2002

Quarter	Sales revenue
Q1	$3,326,172
Q2	$2,840,651
Q3	$2,879,303
Q4	$4,186,120

6 On the **Reporting** toolbar, click the **Variable Editor** button.
The Variable Editor dialog box appears.



7 In the **Variable Definition** zone, enter a name for the variable you are creating. In this example, type **Quarters**.

8 Verify that the variable qualification is defined as a **dimension**.

9 Using the **Data**, **Functions** and **Operators** tabs, enter the following formula in the **Formula** zone:

```
=Replace([Quarter] ; "Q" ; "Quarter ")
```

where:

Syntax	Description
[Quarter]	the name of the object that retrieves the data
"Q"	the value stored in the database, followed by 1 through 4 to indicate the first through fourth quarters of the year
"Quarter "	the character string that will replace "Q" in every instance. **Note:** Be sure to add a space after the word Quarter and before the end-quote.

To complete the formula

1 Enter the **Replace** function in the **Formula** zone.
To do this, you can either:

   1 Click the **Operators** tab and double-click the = operator.

   2 Click the **Functions** tab, scroll down the list and double-click the **Replace** function.

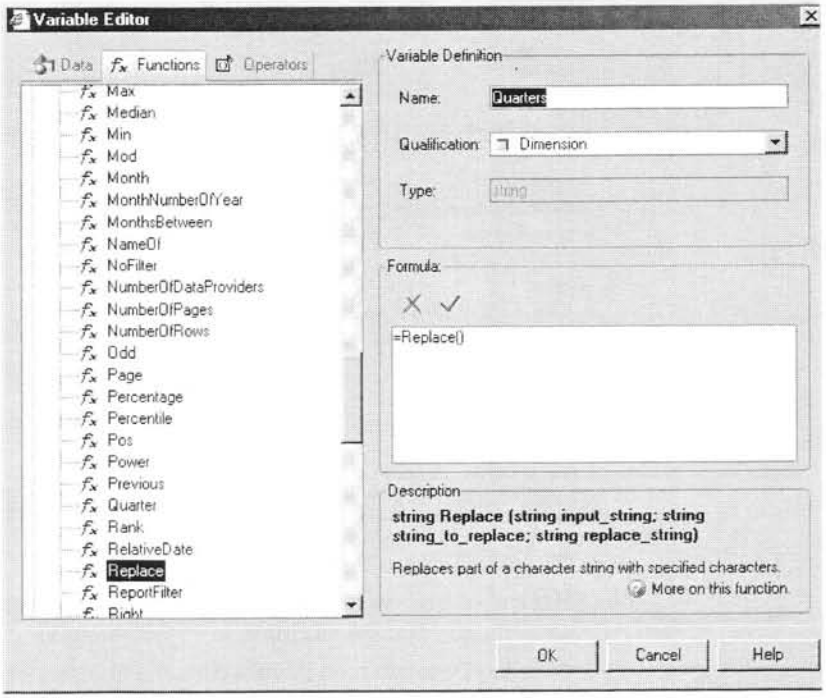
   Or

1 Double-click the **Replace** function so that the function appears in the Formula zone, automatically preceded by the = operator.

Note: Remember that you always begin a formula with the equal (=) sign.

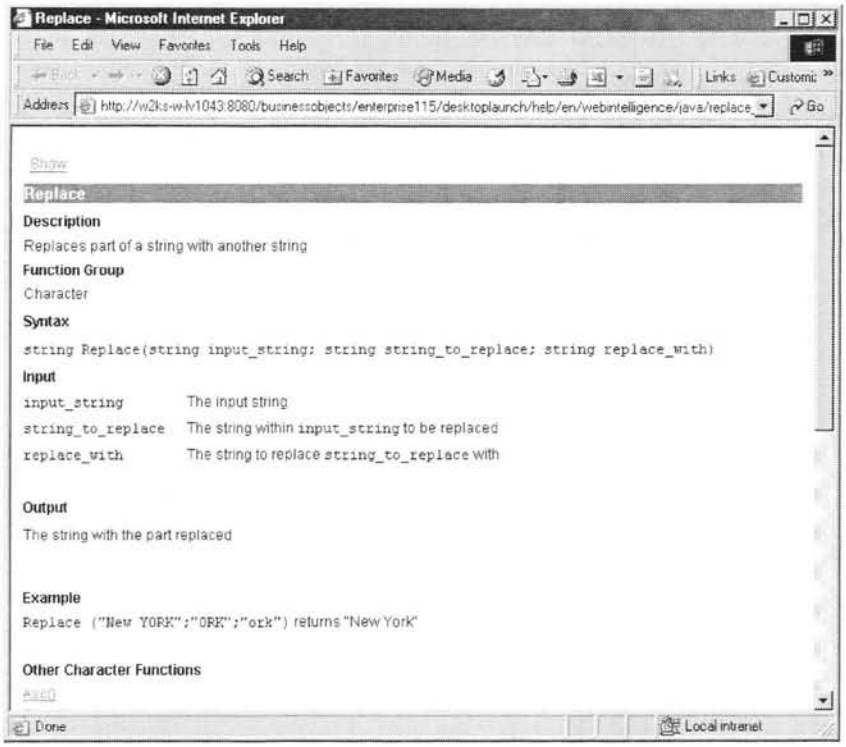
As soon as you enter a function in the Formula zone, a description appears in the bottom zone of the Variable Editor that provides the syntax required for the function that you have selected.

Note that the syntax indicates you must use the semi-colon (;) character to separate the strings in the formula.



2 Click the **More on this function** link to get help on using this function.

A Help window appears that displays a description of the function, the syntax required to use the formula and an example of how it can be used.

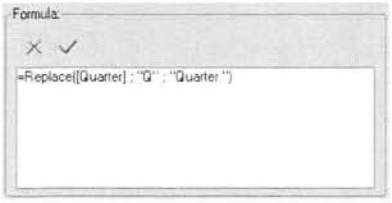


3 Click the **Close** icon to close this Help window.

4 Make sure that your cursor is still positioned between the open and close parentheses that follow Replace, in the formula.

5 Click the **Data** tab then double-click the **Quarter** object.
The Quarter object appears in the formula.

6 Complete the formula using the appropriate character strings, as shown here:



7 Click the **Validate** button in the **Formula** zone, to validate that you have used the correct syntax.

8 If the syntax is correct, click **OK** to close the Variable Editor.
The new variable now appears in the list of objects and variables in the Edit Report view Data tab.

To display the data calculated by the new variable in the table

1 From the **Data** tab, drag **Quarters** to the table and drop it over the **Quarter** column, to replace that column with the new variable.
The table now appears like this:

2001

Quarters	Sales revenue
Quarter 1	$2,660,700
Quarter 2	$2,279,003
Quarter 3	$1,367,841
Quarter 4	$1,788,580

2002

Quarters	Sales revenue
Quarter 1	$3,326,172
Quarter 2	$2,840,651
Quarter 3	$2,879,303
Quarter 4	$4,186,120

2 Save this document in your Favorites folder as **Doc_Functions**.

Using the Right() function

Introduction

The Right() function is used to extract a fixed number of characters starting from the right end of the character string.

The syntax for the Right() function is:

```
Right([object] ; number)
```

String refers to the string from which the substring is to be extracted. This is normally the name of an object (for example, [Year]).

Number sets the number of characters to be extracted from the right of the total string length.

After completing this unit, you will be able to:

- Use the Right() function to delete characters from a character string

How to extract a substring using the Right() function

Example

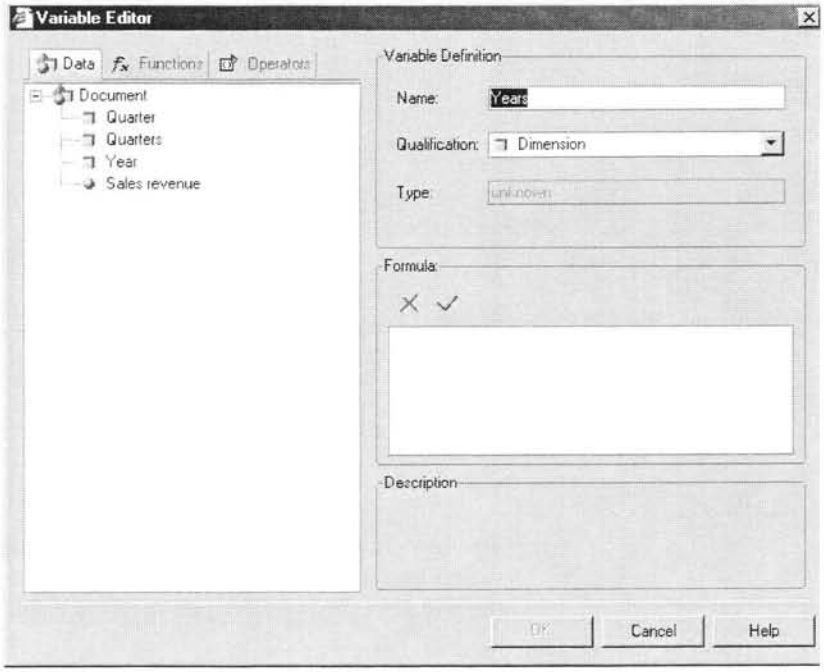
The Year object in the eFashion universe is formatted to return the year as a four-digit number (for example, 2001). However, in our example report you need to display the year in two digit format (for example, 01). To do this, use the Right() character string function.

To use the Right() function

1 Continue using the **Doc_Functions.rep** document in the Java Report Panel.

2 In Edit Report view, click the **Variable Editor** button on the **Report** toolbar.
The Variable Editor dialog box appears.

3 In the **Variable Definition** zone, enter a name for the variable you are creating. In this example, type **Years**.



4 Verify that the variable qualification is defined as a **dimension**.

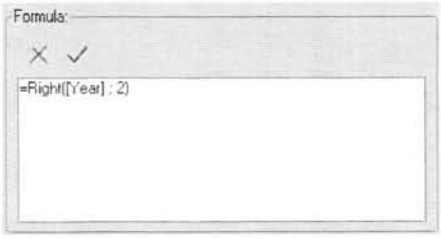
5 Using the **Data**, **Functions** and **Operators** tabs, enter the following formula in the **Formula** zone:

```
=Right([Year] ; 2)
```

where:

Syntax	Description
[Year]	The name of the object that retrieves the data
2	The number of characters to be extracted to the right of the character string

The Formula zone should appear like this:



6 Click the **Validate** button in the **Formula** zone, to validate that you have used the correct syntax.

7 If the syntax is correct, click **OK** to close the Variable Editor.
The new variable now appears in the list of variables in the Variables folder on the Data tab.

8 Drag the variable to the block and drop it in the desired position.
In this instance, swap the **Year** section header with the new variable **Years**.

01

Quarters	Sales revenue
Quarter 1	$2,660,700
Quarter 2	$2,279,003
Quarter 3	$1,367,841
Quarter 4	$1,788,580

02

Quarters	Sales revenue
Quarter 1	$3,326,172
Quarter 2	$2,840,651
Quarter 3	$2,879,303
Quarter 4	$4,186,120

Note: The Left character string function works in exactly the same way except that the function returns the specified number of characters starting from the left as opposed to the right.

9 Save the document.

Using the SubString() function

Introduction

The SubString() function extracts a specified character string from within a string.

The syntax for the Substr() function is:

```
Substr([object] ; position ; number)
```

String is the string from which the substring is to be extracted. This is normally the name of an object (for example, [Store name]).

Position is the position of the first character in the string to be extracted (for example, 1).

Number is the number of characters (from the initial position forward) to be extracted.

After completing this unit, you will be able to:

- Use the SubString() function to extract a character string from within a string

How to extract a string using the SubString() function

Example

The Store name object in the eFashion universe returns the name of each store in the company, but prefaces each one with "e-Fashion." You want to remove the "e-Fashion" portion of the character string.

This presents a challenge since the length of each store's name is different, with or without the 'e-Fashion' portion. Neither the Right() or Left() functions will work here.

Store name
e-Fashion Los Angeles
e-Fashion San Francisco
e-Fashion Colorado Springs
e-Fashion Washington Tolbooth
e-Fashion Miami Sundance
e-Fashion Chicago 33rd
e-Fashion Boston Newbury
e-Fashion New York 5th
e-Fashion New York Magnolia
e-Fashion Austin

The easiest way to achieve the removal is to use the SubStr() function.

To use the SubStr() function

1 Using the eFashion universe, create a new Web Intelligence document.

2 Build a query in the Java Report Panel using the **State, Store name, Quantity sold, Sales revenue** and **Margin** objects.

3 Click **Run Query**.

4 Open the **Variable Editor** and create a new variable called **Store**.

5 Inside the **Formula** zone of the Variable Editor, use the SubStr() function to force this new variable to return ONLY the store names without the 'e-Fashion' preface.

Consider the syntax:

```
=SubStr([object]; beginning number ;number of characters)
```

For our scenario, enter:

```
=SubStr([Store name] ; 11 ; Length([Store name]))
```

Why?

- "11" represents the starting point at which the formula is to start displaying characters. This was established by counting the number of characters in "e-Fashion" (9) and then adding a character to account for the space between the word "e-Fashion" and each store name (10).
- The Length() function represents the number of characters you want returned from the string. The object name (Store name, in this case) determines the length of the string, so if you take the beginning number (11 in this case) from the length of the string, you are instructing the system to return all but the first 11 characters.

Note: Do not forget that the beginning number in the SubStr() function is the beginning character position that you want the substring to start displaying characters. Character positions identified in the SubStr() syntax are those you want to see displayed, not those you want to remove.

6 Replace the **Store name column** in the table with the new **Store** variable so that your table looks like this example:

State	Store	Quantity sold	Sales revenue	Margin
California	Los Angeles	26,244	$4,220,929	$1,668,395
California	San Francisco	19,830	$3,258,641	$1,304,515
Colorado	Colorado Springs	12,787	$2,060,275	$808,149
DC	Washington Tolbooth	18,744	$2,961,950	$1,153,001
Florida	Miami Sundance	11,267	$1,879,158	$777,281
Illinois	Chicago 33rd	17,976	$3,022,658	$1,254,093
Massachusetts	Boston Newbury	7,676	$1,283,707	$511,684
New York	New York 5th	18,094	$2,960,367	$1,201,876
New York	New York Magnolia	28,264	$4,621,854	$1,870,868
Texas	Austin	17,078	$2,699,673	$1,060,310
Texas	Dallas	12,365	$1,970,034	$754,862
Texas	Houston	13,916	$2,303,183	$939,226
Texas	Houston Leighton	18,988	$3,144,774	$1,282,680

7 Save the document in your Favorites folder as **Doc_Marginperformance**.

Using the Pos() function

Introduction

The Pos function returns the number of characters from the start of a string to a specified character.

The syntax for this function is:

```
Pos([object] ; "search string")
```

Search defines the string to be searched for the occurrence of a character and it's position in the string. This is often the name of an object (for example, [Store name]).

Object is the character string you wish to search for.

After completing this unit, you will be able to:

- Use the Pos() function to return only certain characters from the start of a string to a specified character

How to use the Pos() function

Example

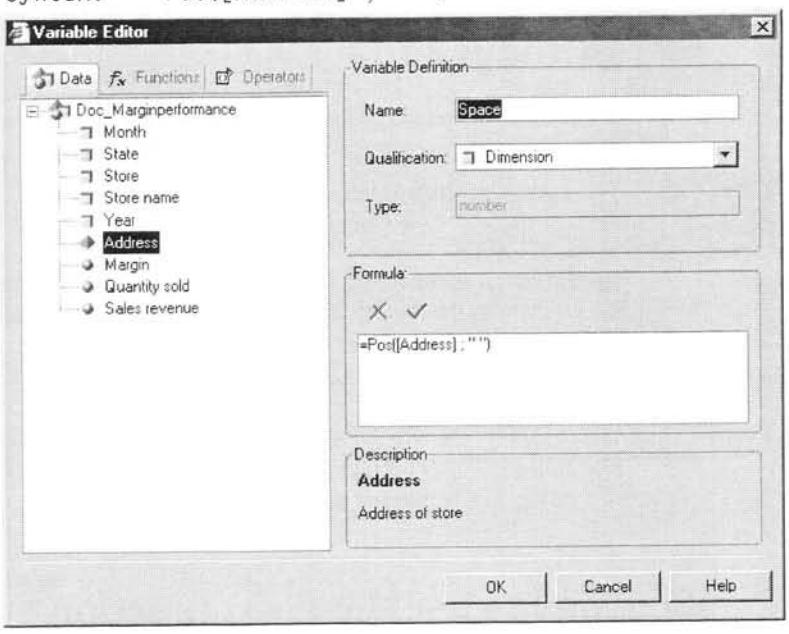
Now you want to display only the street names of each of the eFashion stores. The Address object from the universe returns both the building number and the street name. Therefore, a substring must be created to return only the portion that is needed.

To use the Pos() function

1. Continue using the **Doc_Marginperformance** document.
2. Click **Edit Query** in the Java Report Panel and edit the query to add the **Address** detail object.
3. Click **Run Query**.
4. In Edit Report view, drag the **Address** object from the **Data** tab to insert a new column in the table, to the right of the Store column.
   The address of each store is inserted in a new column.
   Notice that there is a space after each building number. If the formula recognized at what position the space lies for each value, it could use that position as a starting point for the SubStr() function.
   The Pos() function enables you to determine where the space sits and therefore where to begin the substring.
5. Click the **Variable Editor** button on the **Report** toolbar to create a new variable.
6. Name the new variable **Space**.

7 In the **Formula** zone of the Variable Editor, use the **Pos()** function to locate the position of the space in each address.

```
Syntax:   =Pos([Address] ; " ")
```



8 Validate the syntax and click **OK**.

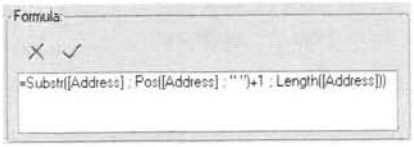
9 Add another new variable using the **Variable Editor**, called **Store Address**.

10 Build this new variable to return only the street name of the store's location. Use a combination of the **SubStr()** function and the new **Space** object.

The syntax should be as follows:

```
=SubStr([Address] ; [Space]+1 ; Length([Address]))
```

You can also use:



Why?

- The Space object sits at the 'beginning number' slot of the SubStr() function's syntax. It contains the variable name and the text pattern you want to locate (a space in this case).
- The +1 tells the function to begin the substring at one position more than the location of the space so that the space is not returned.
- The Length() function instructs the system that the number of characters to be displayed from each Address string is equal to the length of each address after the character (a space in this case) in the beginning number position.

11 Replace the old **Address column** with the new **Store Address** variable so that the table looks like this example:

State	Store	Store Address	Quantity sold	Sales revenue	Margin
California	Los Angeles	Princes Street	26,244	$4,220,929	$1,668,395
California	San Francisco	Ocean Lane	19,830	$3,258,641	$1,304,515
Colorado	Colorado Springs	Wallace Road	12,787	$2,060,275	$808,149
DC	Washington Tolbooth	Tolbooth Street	18,744	$2,961,950	$1,153,001
Florida	Miami Sundance	Sundance Place	11,267	$1,879,158	$777,281
Illinois	Chicago 33rd	33rd Street	17,976	$3,022,658	$1,254,093
Massachusetts	Boston Newbury	Newbury Avenue	7,676	$1,283,707	$511,684
New York	New York 5th	5th Avenue	18,094	$2,960,367	$1,201,876
New York	New York Magnolia	Magnolia Street	28,264	$4,621,854	$1,870,868
Texas	Austin	Forres Street	17,078	$2,699,673	$1,060,310
Texas	Dallas	Bremner Way	12,365	$1,970,034	$754,862
Texas	Houston	Loredo Avenue	13,916	$2,303,183	$939,226
Texas	Houston Leighton	Leighton Place	18,988	$3,144,774	$1,282,680

12 Save the document.

Concatenating different character strings

Introduction

The character used to link two strings together in a formula, or concatenate data, is the plus symbol "+". The syntax is:

```
"String1" + "String2"
```

Note: You can concatenate as many strings together as you like.

If you want to concatenate a number or a date with a string you must first convert the number or date to a string. To convert a date to a string you can use the FormatDate function. The syntax for this function is:

```
FormatDate([date] ; "stringformat")
```

For example:

```
FormatDate([Start Date] ;"dd/mm/yyyy")
```

To convert a number to a string you can use the FormatNumber function. The syntax for the function is:

```
FormatNumber([number] ; "stringformat")
```

After completing this unit, you will be able to:

- Combine two strings in a formula using the concatenate function

How to concatenate a string with a date

To explore concatenating different data types, consider combining a string value with the Last Refresh Date free-standing cell formula.

The Last Refresh Date free-standing cell displays the date and time that the document was last executed in a cell in the report:

1/6/06 4:32 PM

The syntax for the formula of the cell is:

```
LastExecutionDate()
```

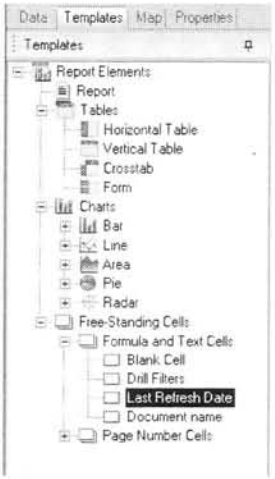
First, you insert the Last Refresh Date free-standing cell into an existing report. To insert a descriptive string prior to the date you concatenate the two.

To concatenate a string with a date

1 Continue using the **Doc_Marginperformance** document.

2 In Edit Report view, click the **Templates** tab then click + to expand the **Free-Standing Cells** templates folder.

3 Click + to expand the **Formula and Text Cells** templates folder.



4 Drag the **Last Refresh Date** cell template and drop it in a blank area of the report, to insert a new cell.

A new cell automatically appears in the report, and the cell displays the date and time that the queries in the document were last executed.

5 Click the **Last Refresh Date** cell in the report to highlight it.

1/6/06 4:32 PM

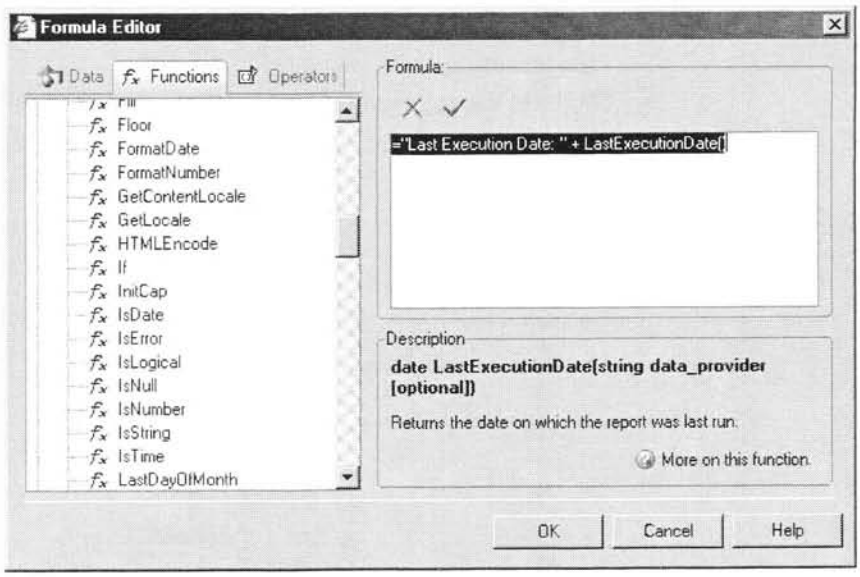
State	Store	Store Address	Quantity sold	Sales revenue	Margin
California	Los Angeles	Princes Street	26,244	$4,220,929	$1,688,395
California	San Francisco	Ocean Lane	19,830	$3,258,641	$1,304,515
Colorado	Colorado Springs	Wallace Road	12,787	$2,060,275	$808,149
DC	Washington Tolbooth	Tolbooth Street	18,744	$2,961,950	$1,153,001
Florida	Miami Sundance	Sundance Place	11,267	$1,879,158	$777,281
Illinois	Chicago 33rd	33rd Street	17,976	$3,022,658	$1,254,093
Massachusetts	Boston Newbury	Newbury Avenue	7,676	$1,283,707	$511,684

6 Click the **Show/Hide Formula Toolbar** button on the **Report** toolbar.

7 On the **Formula** toolbar, click the Formula Editor button.

The Formula Editor appears. Note that the formula used by Last Refresh Date free-standing cell template appears automatically in the Formula zone.

8 Position your cursor just after the equal (=) sign in the **Formula** zone, and insert the string **"Last Execution Date: " +** after the **=** symbol.



9 Validate the syntax of the formula.

10 Click **OK** and resize the cell as appropriate.
It should now appear like this example:

Last Execution Date: 1/25/06

11 Save the document.

Using date functions

Introduction

The three most commonly used date functions are:

- ToDate: changes the data type of a value to date
- CurrentDate: gives the date today
- DaysBetween: calculates the number of days between two dates

In this unit, you will consider the process of converting a string to a date as dates are often formatted as character type at either the database or universe level.

You will use all of these date functions to create a variable named Trading Years.

After completing this unit, you will be able to:

- Use date functions to create a variable
- Convert a character string to a date value

Converting a string to a date value

Example

In the eFashion universe, there is no object for number of years trading. The universe only has the Opening Date dimension object. You want to create a variable that calculates the number of years that stores have been trading, based on the date the stores opened.

To convert a string to a date value

1 Continue using the Doc_Marginperformance document.

2 Click **Edit Query** in the Java Report Panel, to edit the query.

3 Add the **Opening Date** dimension object to the query, from the **Store** class and the **Store details** sub-class.

4 In Edit Report view, drag the **Opening Date** object from the **Data** tab and position it to insert a new column to the right of the **Store address** column.

The table now appears like this:

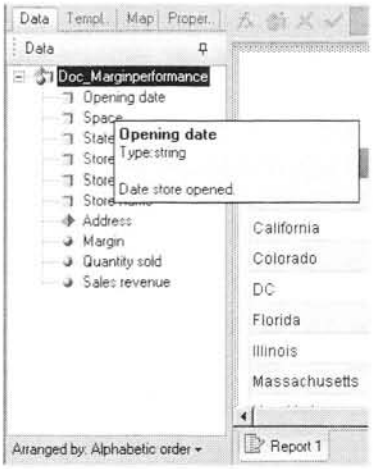
Last Execution Date: 1/25/06

State	Store	Store Address	Opening date	Quantity sold	Sales revenue	Margin
California	Los Angeles	Princes Street	3/22/98 12:00:00 AM	26,244	$4,220,929	$1,
California	San Francisco	Ocean Lane	3/21/98 12:00:00 AM	19,830	$3,258,641	$1,
Colorado	Colorado Springs	Wallace Road	11/3/84 12:00:00 AM	12,787	$2,060,275	$
DC	Washington Tolbooth	Tolbooth Street	12/5/98 12:00:00 AM	18,744	$2,961,950	$1,
Florida	Miami Sundance	Sundance Place	2/23/94 12:00:00 AM	11,267	$1,879,159	$
Illinois	Chicago 33rd	33rd Street	3/14/96 12:00:00 AM	17,976	$3,022,658	$1,
Massachusetts	Boston Newbury	Newbury Avenue	11/21/97 12:00:00 AI	7,676	$1,283,707	$
New York	New York 5th	5th Avenue	5/13/85 12:00:00 AM	18,094	$2,960,367	$1

Note: The values returned by the Opening Date object are displayed in the format "mm/dd/yy hh:mm:ss AM."

5 In the Edit Report view **Data** tab, position your cursor next to the **Opening Date** object.

A pop-up help box appears, providing details about the object you have pointed to. In this case, you can see that the data type of the **Opening Date** object is the character-string type.



Before you can calculate the trading years you need to strip out the time element of the string and then convert it to a date type.

Use the Substr() function to strip out the time element of the string. However, the date element is not in a constant format (for example, some days and months are expressed as single digits while others are two digits).

Since the date is always followed by a space you can use the Pos() function to define the number of characters element of the Substr() syntax.

6 Create a variable called **Date_Substring** based on this formula.

```
=SubStr([Opening date] ; 1 ; Pos([Opening date] ;" ")-1)
```

7 Create another variable called **Open_Date** to convert the [Date_Substring] values to date type using this formula.

```
=ToDate([Date_Substring] ; "mm/dd/yy")
```

Note: The date must be formatted in the same way as the string that is being converted. For instance, if the year in the string is two digits you must express the date format as two digits.

8 Save the document.

Using date calculations

Now that you have created a date type variable for the store opening date, you can calculate the number of days to the current date in days using the CurrentDate and DaysBetween functions. The syntax of the DaysBetween function is:

```
Daysbetween(date 1 ; date 2)
```

Date 1: The oldest date

Date 2: The most recent date

To create a date calculation

1 Continue using the document from the previous procedure.

2 Create a variable called **Trading_Years** based on this formula:

```
=DaysBetween([Opendate] ; CurrentDate())/365.25
```

Note: To convert the days to years, you must divide by the number of days in a year and in so doing take account of leap years. Hence, the date calculation is divided by 365.25 days.

3 Drag the new variable **Trading_Years** onto the **Opening date** column in the table.

The Trading_Years variable replaces the Opening Date column in the table block.

Last Execution Date: 1/25/06

State	Store	Store Address	Trading_Years	Quantity sold	Sales revenue	Margin
California	Los Angeles	Princes Street	8.01	26,244	$4,220,929	$1,668,395
California	San Francisco	Ocean Lane	8.01	19,830	$3,258,641	$1,304,515
Colorado	Colorado Springs	Wallace Road	22.06	12,787	$2,060,275	$808,149
DC	Washington Tolbooth	Tolbooth Street	8.05	18,744	$2,961,950	$1,153,001
Florida	Miami Sundance	Sundance Place	12.01	11,267	$1,879,159	$777,281
Illinois	Chicago 33rd	33rd Street	10.03	17,976	$3,022,658	$1,254,093
Massachusetts	Boston Newbury	Newbury Avenue	9.01	7,676	$1,283,707	$511,684
New York	New York 5th	5th Avenue	21.03	18,094	$2,960,367	$1,201,676
New York	New York Magnolia	Magnolia Street	8	28,264	$4,621,854	$1,870,868
Texas	Austin	Forres Street	24.06	17,078	$2,699,673	$1,060,310

Note: The results in your table will differ from the example shown as it displays the date of execution.

4 Change the new data column to the **Number format** of your choice.

5 Save the document as **Doc_Marginperformance2**.

Practice

Activity: Character and Date String Functions

Objectives

Use the SubStr() functions in conjunction with the Length() function to do high-level string formatting.

Instructions

Create a report showing the margin performance across all eFashion stores.

1 Create a report using the eFashion universe and select the following objects:
State, Store name, Quantity Sold, Sales Revenue, Margin

2 Apply a sum to all measures.

3 There is a lot of repetition in the Store name column. Create a variable called Store to remove the eFashion from the Store name.

4 Save the report as **Act_characterdatestring.rep**.
Your report should now look like this:

State	Store	Quantity sold	Sales revenue	Margin
California	Los Angeles	26,244	$4,220,929	$1,668,395
California	San Francisco	19,830	$3,258,641	$1,304,515
Colorado	Colorado Springs	12,787	$2,060,275	$808,149
DC	Washington Tolbooth	18,744	$2,961,950	$1,153,001
Florida	Miami Sundance	11,267	$1,879,159	$777,281
Illinois	Chicago 33rd	17,976	$3,022,658	$1,254,093
Massachusetts	Boston Newbury	7,676	$1,283,707	$511,684
New York	New York 5th	18,094	$2,960,367	$1,201,876
New York	New York Magnolia	28,264	$4,621,854	$1,870,868
Texas	Austin	17,078	$2,699,673	$1,060,310
Texas	Dallas	12,365	$1,970,034	$754,862
Texas	Houston	13,916	$2,303,183	$939,226
Texas	Houston Leighton	18,988	$3,144,774	$1,282,680
	Sum:	**223,229**	**$36,387,203**	**$14,586,940**

Lesson summary

Review

Quiz: Character and Date String Functions

1 Name at least three character string functions used in this lesson.

2 To find the occurrence of a comma in an object's value, which function would you use?

3 What is the syntax for the Replace() function?

4 What does *concatenation* mean?

5 Give an example of something concatenated.

6 What happens if you put text and a date together in a variable?

7 How do you resolve this?

8 Can you do calculations on a date?

Summary

After completing this lesson, you are now able to:

- Describe the character-string functions presented in this lesson
- Use the Replace() function to replace "Q" with the word "Quarter"
- Use the Right() function to delete characters from a character string
- Use the SubString() function to extract a character string from within a string
- Use the Pos() function to return only certain characters from the start of a string to a specified character
- Combine two strings in a formula using the concatenate function
- Use date functions to create a variable
- Convert a character string to a date value

Lesson 3
Using If Logic

This lesson shows you how to use the If() function formula in a variety of ways, including how to group values in a variable, and how to modify the way calculations behave when certain values are returned by an object.

In this lesson you will learn about:

- Grouping data
- Using the If() function to modify calculation behavior

Duration: 1 hour

Grouping data

Introduction

The If() function is a powerful tool available to Web Intelligence report designers. Also referred to as "If...Then...Else" logic, the If() function is commonly used to group values returned from the database into categories. Using the If() in this manner, report designers are empowered to perform the report equivalent of the Decode, Case, and If functions available to universe designers at the database level.

If() can be used to group some of the values returned by an object, and then define that grouping as a new variable in the document.

After completing this unit, you will be able to:

- Group values using the If() function
- Define the grouping as a new variable in the document

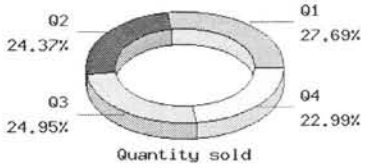
Grouping values with the If() function to show higher levels of details

You can use the If() function in Web Intelligence to group values returned by an object into categories. These categories can then be used as the basis for aggregating measures at higher levels of detail.

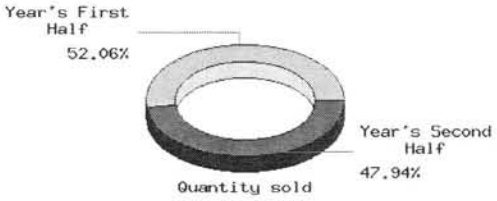
For example, the eFashion universe contains Quarter and Quantity Sold objects, which allow you to analyze the quantity sold totals for each quarter. However, if your reporting requirement is to display the quantity sold measure aggregated at the half year level of detail, then the universe does not provide the objects you need to meet this requirement.

By creating a Half Year variable based on the Quarter object, you can aggregate quantity sold totals for each half of the year. This data can then be presented in the same report with quantity sold totals for each quarter.

Quarter	Quantity sold
Q1	61,808
Q2	54,406
Q3	55,690
Q4	51,325
Sum:	223,229



Half Year	Quantity sold
Year's First Half	116,214
Year's Second Half	107,015
Sum:	223,229



In this example, the Half Year variable looks at each value for the Quarter. If the quarter's returned value is either Q1 or Q2 then the variable displays the words "Year's First Half" and if the quarter's value is anything other than Q1 or Q2 then it will display the words "Year's Second Half".

The syntax used with the If() function is:

```
=If([object name] = "true value"; "value to display if
true"; "value to display if false")
```

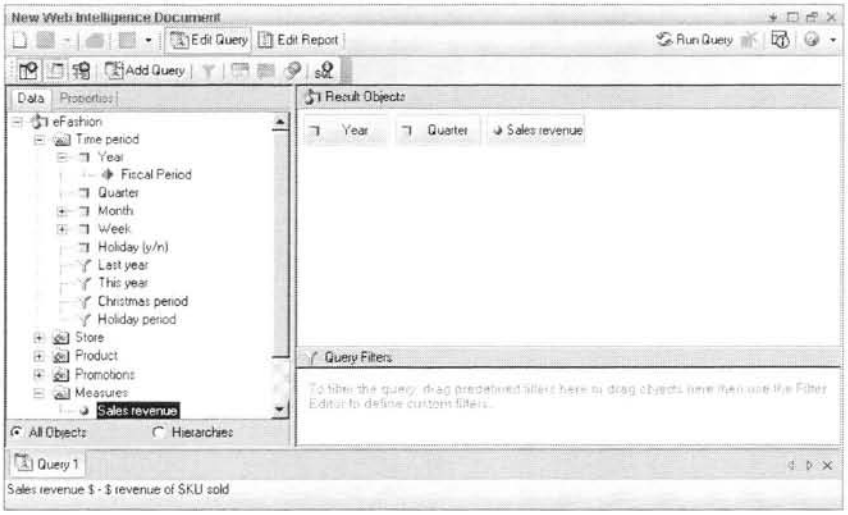
Scenario

You want to show sales revenue broken down to the half-year level. Since no object exists for half year in the eFashion universe, you will create the formula and define it as a variable in the document.

To group values using the If() function in a formula

1 Create a new Web Intelligence document using the eFashion universe.

2 In the Edit Query view of the Java Report Panel, drag or double-click the **Year**, **Quarter** and **Sales revenue** objects to move them into the Result Objects pane.



3 Click **Run Query**.

You will now use the values of the Quarter object to generate the needed Half Year groups:

Quarters = Half Years

Q1 + Q2=Year's First Half

Q3 + Q4=Year's Second Half

4 In Edit Report view, click the **Variable Editor** button on the **Reporting** toolbar.

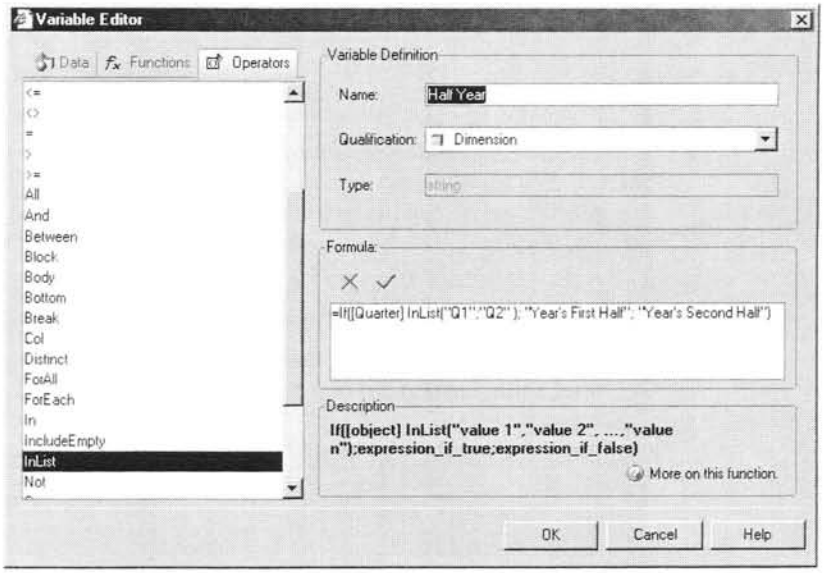
5 In the **Variable Editor** dialog box, create a new variable called **Half Year.**

6 Verify that the variable is qualified as a dimension.

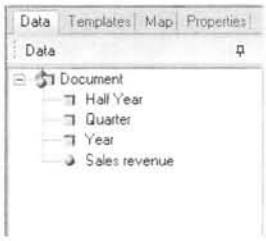
7 Build the variable's formula using this syntax:

```
=If([Quarter] InList("Q1";"Q2" ); "Year's First Half";
"Year's Second Half")
```

8 Click the **Validate** button to check your syntax.
The Variable Editor appears as below:



9 Click **OK** to create the new variable and close the Variable Editor.
The new variable appears in the Data tab in Edit Report view.



10 Drag the new variable and drop it to the right of the **Year** column.

11 Remove the **Quarter** column from the table.

12 Place a **break** on **Year**.
The table appears as shown here:

Year	Half Year	Sales revenue
2001	Year's First Half	$4,939,703
	Year's Second Half	$3,156,421
2001		

Year	Half Year	Sales revenue
2002	Year's First Half	$6,166,823
	Year's Second Half	$7,065,423
2002		

Year	Half Year	Sales revenue
2003	Year's First Half	$7,749,706
	Year's Second Half	$7,309,436
2003		

13 Save this document to your Favorites folder as **Doc_IfGroup**.

Grouping values with the If() function to show categories

Your reporting requirement is to display revenue totals in both a table and a 3D pie chart aggregated for each of the eFashion Market Types. The three market types are Major Market, Mid Market and Minor Market. Each state will need to be assigned to the appropriate market type based on the following business rules:

Markets	States
Major Market	California, New York, Texas
Mid Market	Colorado, DC, Illinois
Minor Market	Florida, Massachusetts

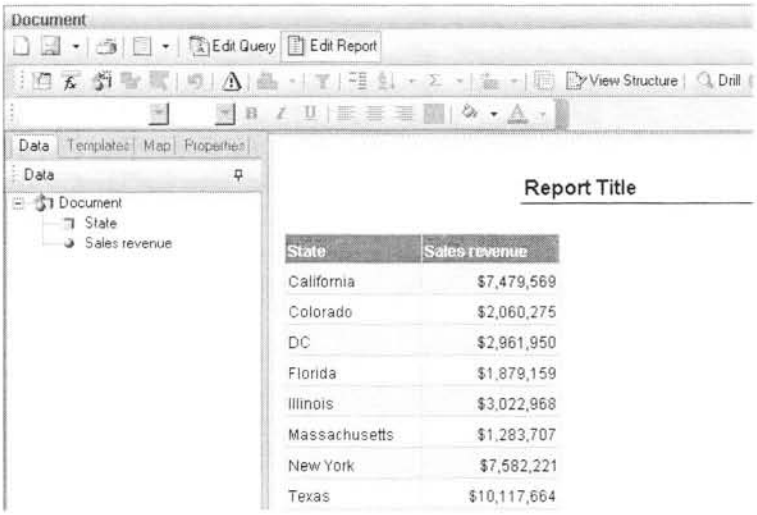
You will need to display state revenue figures including market-type subtotals in the table and market-type revenue percentage breakdowns in the chart.

To use the If() function to show categories of values

1 Create a new Web Intelligence document using the eFashion universe.

2 In Edit Query view, drag or double-click the **State** and **Sales revenue** objects to move them to the Result Objects pane.

3 Click **Run Query.**
The report appears in Edit Report view.



4 Click the **Variable Editor** button on the **Reporting** toolbar.

5 In the Variable Editor dialog box, name the new variable **Market Type** and verify that the variable is qualified as a **dimension**.

6 Click the **Functions** tab and double-click the **If** function to insert it in the formula.
The If function is inserted in the Formula zone, automatically preceded by the = sign.

7 Check that your cursor is inserted between the parentheses that follow the If function.

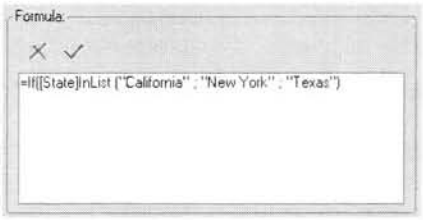
8 Click the **Data** tab, and double-click the **State** object to insert it between the parentheses in the formula.

9 Click the **Operators** tab, and then double-click the **InList** operator.

10 Type a single parentheses **(** character.

11 Begin typing the values that will define the Major Market states. Remember to enclose each state in double-quotes and separate the values with a semi-colon.

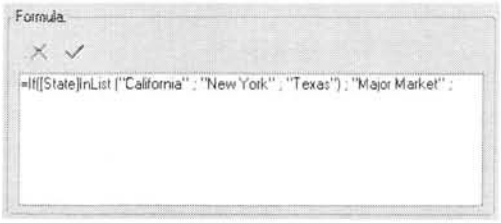
12 Type **)** to close the parentheses.
The formula now appears like this:



13 Position your cursor to the far right of the formula. Type a semi-colon **;** and then **"Major Market"**, followed by a semi-colon **;**.

Note: Remember to include the double-quotes, and to separate all expressions with a semi-colon.

The unfinished formula now appears like this:

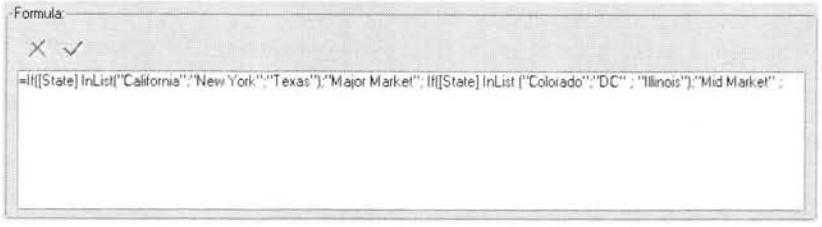


Now that you have defined the three states that fall into the Major Market category, you will define the states that fall into the Mid Market category.

14 Following the same procedure, continue defining **Mid Market** category, using the following syntax:

```
If([State]InList("Colorado" ; "DC" ; "Illinois") ; "Mid
Market";
```

The unfinished formula now appears like this:



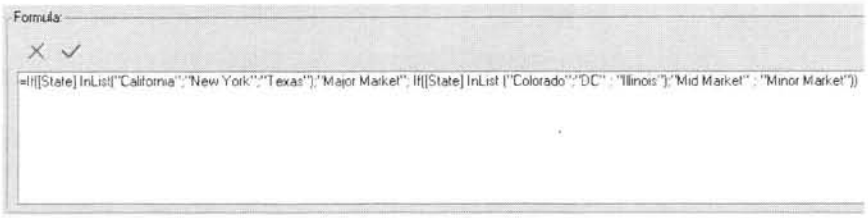
Now that you have defined the states that fall into the Major and Mid Market categories, you will define the states that fall into the Minor Market category.

Because all *remaining* states fall into this third Market Type, you can simply handle the grouping by specifying that Minor Market is the value to assign to any state that does not fall into the first two Market Types.

15 Complete the formula using the following syntax:

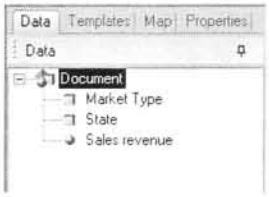
```
"Minor Market"))
```

The completed formula now appears like this:



16 Click the **Validate** button to check your syntax.

17 Click **OK** to create the variable and close the Variable Editor.
The new variable appears in the Data tab in Edit Report view:

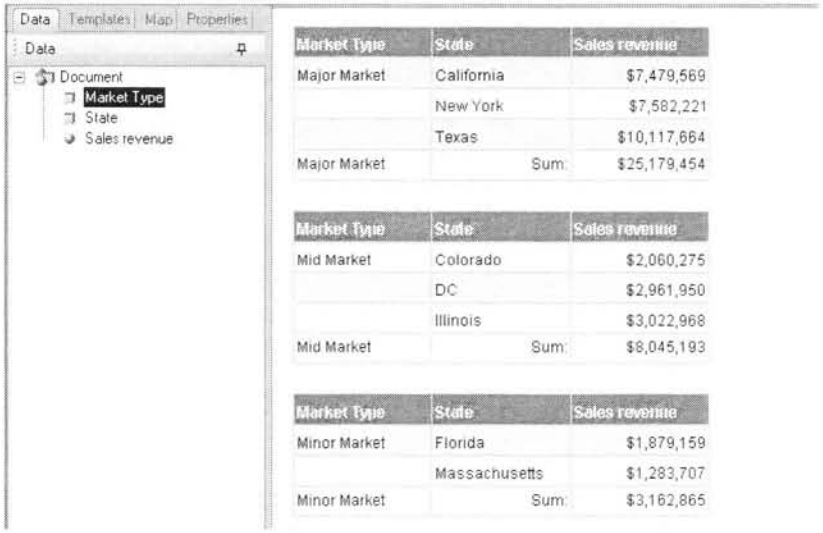


To display the grouped data in the report

1 Drag the new **Market Type** variable from the **Data** tab in Edit Report view and drop it to the left of the **State column** in the existing table.

2 Select the **Market Type** column in the table and click the **Insert/Remove Break** button on the **Reporting** toolbar.

3 Select the **Sales revenue** column and click the **Insert Sum** button on the **Reporting** toolbar.
The report appears as shown here:



Market Type	State	Sales revenue
Major Market	California	$7,479,569
	New York	$7,582,221
	Texas	$10,117,664
Major Market	Sum:	$25,179,454

Market Type	State	Sales revenue
Mid Market	Colorado	$2,060,275
	DC	$2,961,950
	Illinois	$3,022,968
Mid Market	Sum:	$8,045,193

Market Type	State	Sales revenue
Minor Market	Florida	$1,879,159
	Massachusetts	$1,283,707
Minor Market	Sum:	$3,162,865

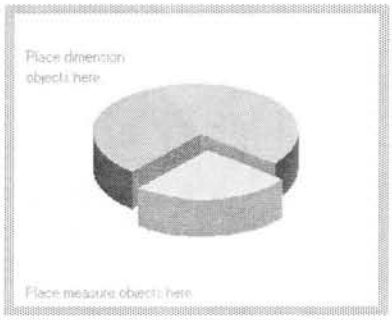
To display the grouped values in a chart

1 In Edit Report view, click the **Templates** tab.

2 In the **Charts** templates folder, expand the **Pie** folder.

3 Drag the **3D Pie Doughnut** template and drop it to the right of the existing table in the report.
The Java Report Panel switches to View Structure mode, so that you can swap objects in the chart without actually displaying the data.

=NameOf([Market	=NameOf([State])	=NameOf([Sales re
=[Market Type]	=[State]	=[Sales revenue]
=[Market Type]		Sum: m([Sales revenue])
		Sum: m([Sales revenue])



4 Click the **Data** tab.

5 Drag the **Market Type** variable and drop it where the chart template displays *Place dimension objects here.*

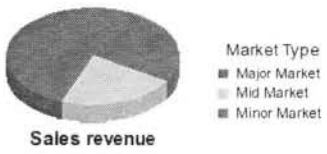
6 Drag the **Sales revenue** object and drop it where the chart template displays *Place measure objects here.*

7 Click the **View Results** button on the **Reporting** toolbar.
The pie chart appears beside the existing table. You can now edit the chart as you choose, by displaying the revenue values as percentages or real numbers.

Market Type	State	Sales revenue
Major Market	California	$7,479,569
	New York	$7,582,221
	Texas	$10,117,664
Major Market	Sum:	$25,179,454

Market Type	State	Sales revenue
Mid Market	Colorado	$2,060,275
	DC	$2,961,950
	Illinois	$3,022,968
Mid Market	Sum:	$8,045,193

Market Type	State	Sales revenue
Minor Market	Florida	$1,879,159
	Massachusetts	$1,283,707
Minor Market	Sum:	$3,162,865



Sales revenue

Practice

Activity: Grouping data

Objective

- Use the If() function syntax to group data values

Instructions

You want to group store managers by their degree of experience managing their store. Then, you are going to create a report that shows the sales revenue earned by each manager in each tenure-type category.

It is necessary to create a variable to group the managers into categories, because experience, or tenure type, is not an object that is available in the eFashion universe.

1 Create a new Web Intelligence document using the **eFashion** universe.

2 Include the **Name of manager, State**, and **Sales revenue** objects in the query.

3 Create a variable called **Tenure type**.

4 Qualify the variable as a **dimension**.

5 Define the formula to group each store manager into a category, based on the following tenure-type categories:

Tenure type	Name of manager
Rookies	Bennett
	Leonard
	Tuttle
Subs	Queen
Veterans	Anderson
	Barrett
	Larry
	Mark
	Michelle
	Quinn
	Richards
	Steve

6 Insert the Tenure type variable into the table so that its values are projected in the first column in the table.

7 Format the report so that it is organized in blocks that show the sales revenue earned by the managers in each tenure-type category. Your report should look like this example:

Tenure type	Name of manager	State	Sales revenue
Rookies	Bennett	Colorado	$2,060,275
	Leonard	Texas	$1,970,034
	Tuttle	Florida	$1,879,159
Rookies			

Tenure type	Name of manager	State	Sales revenue
Subs	Queen	Texas	$2,303,183
Subs			

Tenure type	Name of manager	State	Sales revenue
Veterans	Anderson	New York	$4,621,854
	Barrett	DC	$2,961,950
	Larry	Texas	$2,699,673
	Mark	Massachusetts	$1,283,707
	Michelle	Texas	$3,144,774
	Quinn	Illinois	$3,022,658
	Richards	New York	$2,960,367
	Steve	California	$7,479,569
Veterans			

8 Save the document in your Favorites folder as **Act_Groupdata**.

Using the If() function to modify calculation behavior

Introduction

The If() function is useful for extending the functionality and flexibility of calculations in reports. More specifically, by using the If() function, report designers can change how a measure behaves based on each value returned for an object.

After completing this unit, you will be able to:

- Use the If() function to modify how data is calculated depending on the values returned by an object used in the report

Modifying the way calculations behave

You can also use the If() function to extend the functionality of calculations so that they behave more dynamically.

The syntax for using the If() function to modify the behavior of calculations is:

```
=If([object name]= "true value"; [measure] behavior A ;
[measure] behavior B)
```

In other words: if the value returned by an object is equal to what is specified in quotes, then the calculation should behave in a certain manner; otherwise, the value should be calculated in a different manner.

Scenario

You want to build a report that calculates the target revenue for each Store Name. Each store has a different percentage of growth it is supposed to achieve based on the state in which it is located. Therefore, a single calculation will not work.

The stores in the following states are to grow their revenue based on these percentages:

State	% Growth Required
California	5%
Colorado	10%
DC	15%
Florida	15%
Illinois	15%
Massachusetts	18%

State	% Growth Required
New York	15%
Texas	15%

To create a variable that calculates growth in revenue

1 Create a new Web Intelligence document using the eFashion universe.

2 In the Java Report Panel, drag or double-click the **State**, **Store name** and **Sales revenue** objects to move them to the Result Objects pane.

3 Click **Run Query**.
The report appears as shown here:

State	Store name	Sales revenue
California	e-Fashion Los Angeles	$4,220,929
California	e-Fashion San Francisco	$3,258,641
Colorado	e-Fashion Colorado Springs	$2,060,275
DC	e-Fashion Washington Tolbooth	$2,961,950
Florida	e-Fashion Miami Sundance	$1,879,159
Illinois	e-Fashion Chicago 33rd	$3,022,658
Massachusetts	e-Fashion Boston Newbury	$1,283,707
New York	e-Fashion New York 5th	$2,960,367
New York	e-Fashion New York Magnolia	$4,621,854
Texas	e-Fashion Austin	$2,699,673
Texas	e-Fashion Dallas	$1,970,034
Texas	e-Fashion Houston	$2,303,183
Texas	e-Fashion Houston Leighton	$3,144,774

Now you are going to calculate each state's target revenue, based on their expected growth.

4 In Edit Report view, click the **Variable Editor** button on the **Reporting** toolbar.

5 Create a new variable called **Target Revenue Based on Required % Growth**.

6 Check that the variable is qualified as a **measure**.

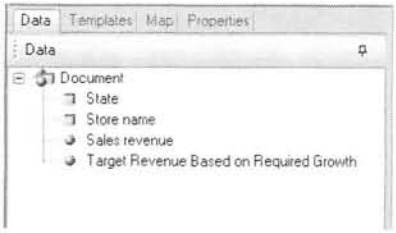
7 Use the If() function to define the syntax of the variable as follows:

```
=If([State]="California" ; [Sales revenue]*1.05 ;
If([State]="Colorado" ; [Sales revenue]*1.1 ;
If([State]InList ("DC";"Florida";"Illinois";"New
York";"Texas ") ; [Sales revenue]*1.15 ;
If([State]="Massachusetts" ; [Sales revenue]*1.18))))
```

Note: When you insert this formula, be sure to include a blank space after Texas and before the closing quote. Texas is entered in the database with a space at the end. Otherwise, the value for Texas will not be calculated in your report.

8 Click the **Validate** button to check your syntax.

9 Click **OK** to close the Variable Editor.
The new variable appears in the Data tab, in Edit Report view.



To calculate the data and project it in the table

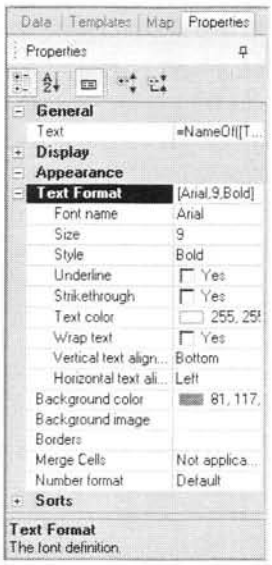
1 Drag the **new variable** and drop it to the right of the **Sales revenue** column, to insert the calculated target revenue data in a new column in the table.

2 Format the new column header cell so that the variable name text is displayed fully in the cell.
To do this:

   1 Click the new **column header cell** to highlight it.

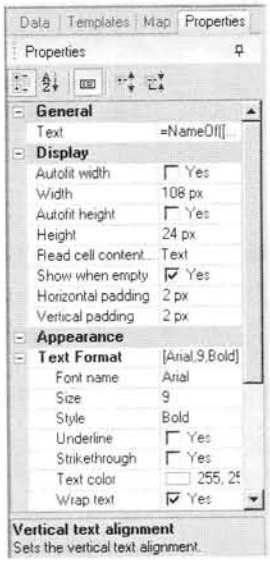
   2 Click the **Properties** tab.

   3 Expand the **Text Format** section in the **Properties** tab.
   The options available for formatting the text in the selected cell appear.



4 In the **Wrap text** option, select **Yes**.

5 Expand the **Display** section in the **Properties** tab.
The options available for formatting the display of the selected cell appear.



6 Select the **Autofit height** option to enlarge the cell so that the full variable name can be seen.

7 Highlight all the column headers in the table by holding down the **Ctrl** key and clicking each **cell** to select them all.

8 In the **Text Format** section of the Properties tab, set the **Vertical Text Alignment** setting to **Center**.

3 Change the formatting of the **Target Revenue values** so that they use the same formatting as the values in the **Sales revenue** column.
To do this:

1 **Right-click** the new column and select the **Format Number** option from the drop-down menu.

2 In the Number Format dialog box, click the **Currency** option.

3 Select the appropriate currency display from the **Properties** list box and click **OK**.

The completed report should look like this example:

State	Store name	Sales revenue	Target Revenue Based on Required Growth
California	e-Fashion Los Angeles	$4,220,929	$4,431,975.24
California	e-Fashion San Francisco	$3,258,641	$3,421,572.53
Colorado	e-Fashion Colorado Springs	$2,060,275	$2,266,302.72
DC	e-Fashion Washington Tolbooth	$2,961,950	$3,406,242.39
Florida	e-Fashion Miami Sundance	$1,879,159	$2,161,032.28
Illinois	e-Fashion Chicago 33rd	$3,022,658	$3,476,057.16
Massachusetts	e-Fashion Boston Newbury	$1,283,707	$1,514,773.79
New York	e-Fashion New York 5th	$2,960,367	$3,404,421.47
New York	e-Fashion New York Magnolia	$4,621,854	$5,315,132.21
Texas	e-Fashion Austin	$2,699,673	$3,104,624.18
Texas	e-Fashion Dallas	$1,970,034	$2,265,539.33
Texas	e-Fashion Houston	$2,303,183	$2,648,660.22
Texas	e-Fashion Houston Leighton	$3,144,774	$3,616,490.22

Note: Even though the Target Revenue Based on Required % Growth variable was calculated using the State object, Web Intelligence is able to display the values at the Store name level, due to the default calculation context of all variables.

4 Save this document in your Favorites folder as **Doc_Modifycalculations**.

Syntax of the If() function

This syntax is based on the If...Then...Else logic and the percentage rates that each state is supposed to grow by.

Some considerations regarding this syntax are:

- When you list a single state, you place the "=" sign before the State value, but when two (or more) states are listed, you use the "InList" operator.
- The syntax requires that when multiple values (States) are listed, all the values must appear in parentheses and each individual value must be in quotes as well as separated from the other values by a semi-colon.

Practice

Activity: Modifying the calculation behavior

Objective

- Use the If() function to create a variable that calculates the sales tax paid by each store, per quarter

Instructions

Using the eFashion universe, you want to display the sales tax paid by each store, and per quarter. You need to create a Sales Tax Paid variable that uses each store's revenue total to calculate sales tax based on the unique tax rate of the state where each store is located.

It is necessary to create a variable to calculate this data, because neither sales tax paid nor each state's tax rate is available as an object in the universe.

1 Create a new Web Intelligence document using the **eFashion** universe.

2 Include the **Quarter**, **State**, **Store name** and **Sales revenue** objects in the query.

3 Create a variable called **Sales Tax Paid**.

4 Qualify the variable as a **measure**.

5 Define the formula to calculate each store's sales tax based on the state where the store is located. Use the table below to specify the tax rate:

State	Tax rate
California	8.5%
Colorado	6.5%
DC	8.5%
Florida	7.5%
Illinois	7.5%
Massachusetts	8.5%
New York	8.5%
Texas	6.5%

6 Format the report so that each block in the report shows the sales revenue and sales tax paid by each store, per quarter.

Your report should look like this example:

Quarter	Store name	Sales revenue	Sales Tax Paid
Q1	e-Fashion Austin	$775,483	$50,406.38
	e-Fashion Boston Newbury	$312,896	$20,338.27
	e-Fashion Chicago 33rd	$846,408	$63,480.63
	e-Fashion Colorado Springs	$525,682	$34,169.34
	e-Fashion Dallas	$555,459	$36,104.82
	e-Fashion Houston	$614,285	$39,928.50
	e-Fashion Houston Leighton	$930,343	$60,472.31
	e-Fashion Los Angeles	$1,129,178	$95,980.10
	e-Fashion Miami Sundance	$515,688	$38,676.58
	e-Fashion New York 5th	$804,084	$68,347.17
	e-Fashion New York Magnolia	$1,183,030	$100,557.58
	e-Fashion San Francisco	$770,503	$65,492.73
	e-Fashion Washington Tolboo	$766,822	$65,179.84
Q1			

Quarter	Store name	Sales revenue	Sales Tax Paid
Q2	e-Fashion Austin	$667,850	$43,410.27
	e-Fashion Boston Newbury	$291,431	$18,943.02
	e-Fashion Chicago 33rd	$850,595	$63,794.63
	e-Fashion Colorado Springs	$509,076	$32,504.04

Lesson summary

Review

Quiz: Using If Logic

1 What is another term for the logic used by the If() function?

2 Which of the following are examples of why you might use the If() function to group values in a variable:

   1 To display categories of values

   2 To combine values from two different objects of the same type

   3 To show values aggregated at higher levels of detail

3 True/False. You can use the If() function to define a variable so that a different calculation is used depending on the value retrieved by the object specified in the variable's formula.

Summary

After completing this lesson, you are now able to:

- Group values using the If() function
- Define the grouping as a new variable in the document
- Use the If() function to modify how data is calculated depending on the values returned by an object used in the report

Lesson 4

Advanced Reporting Techniques

In this lesson you will explore several reporting techniques, including formatting multiple breaks in tables and crosstabs, building a formula to change numeric display of months to text display, creating a custom sort, and displaying the user's response to prompts in free-standing cells.

In this lesson, you will learn about:

- Formatting breaks
- Creating custom sorts
- Displaying document data in free-standing cells
- Displaying data restricted by a filter or ranking

Duration: 1 hour

Formatting breaks

Introduction

Although Web Intelligence allows you to produce reports in several different table formats (vertical, horizontal, crosstab and form), these basic formats do not always fulfill all the requirements that you require in your working environment.

This unit explains several different formatting techniques, using breaks, to display the results of queries in table formats slightly different from the defaults.

After completing this unit, you will be able to:

- Manipulate break headers and footers to change the display of the data
- Prioritize multiple breaks in a table
- Use breaks to format crosstabs

Controlling break headers and footers

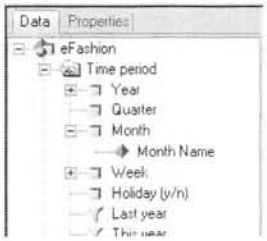
Because breaks are often used as a method of breaking apart table data in order to create sub-totals, one of the most important features of a break is the break footer into which the sub-total calculation is placed. Each time you place a break on a table, you also create a header. Since both the header and footer are created automatically when the break function is utilized, it is important to understand how they operate and how they can be manipulated.

To view default break settings

1 Create a new Web Intelligence document using the **eFashion** universe.

2 Build a query using the **Year**, **Quarter**, **Month Name** and **Sales revenue** objects.

Tip: Click **+** next to the **Month** object to expand the folder and view the **Month Name** detail object in the Data tab.



Because you selected the Month Name detail object, the object that it is associated with, Month, is automatically added to the query as well.

In the eFashion database, the values in the month table are numbers from one to twelve, instead of the full month names. You will see that the Month Name detail object displays the values with their full names.Click **Run Query**.

3 Remove the **Month column** from the table.

4 Using the **Insert Break** button on the **Reporting** toolbar, apply a break on both the **Year** and **Quarter** columns in the table.
The table appears like this:

Year	Quarter	Month Name	Sales revenue
2001	Q1	February	$630,073
		January	$1,003,541
		March	$1,027,085
	Q1		
Year	Quarter	Month Name	Sales revenue
	Q2	April	$895,260
		June	$517,819
		May	$865,615
	Q2		
Year	Quarter	Month Name	Sales revenue
	Q3	August	$173,756
		July	$525,904
		September	$668,181
	Q3		
Year	Quarter	Month Name	Sales revenue
	Q4	December	$649,350
		November	$484,024
		October	$655,206
	Q4		
2001			

In viewing the table, you can see:

- Break header: There is a break on both Year and Quarter, but there are only table headers for each new quarter.
- Break footer: There is a new footer for each new year, as well as each new quarter. If you place a sum on the Sales revenue column, the revenue figures are calculated on both the quarterly and yearly levels and then placed into the appropriate folder.
- The data in the table is currently sorted in the default sort order, that is, in ascending order based on values returned by the measure object included in the query. This means that the months with the lowest revenue appear first, for each quarter.

5 Click the **Sales revenue** column and click the **Insert Sum** button on the Reporting toolbar.

The table appears like this:

Year	Quarter	Month Name	Sales revenue
2001	Q1	February	$630,073
		January	$1,003,541
		March	$1,027,085
	Q1	Sum	$2,660,700
Year	Quarter	Month Name	Sales revenue
	Q2	April	$895,260
		June	$517,819
		May	$865,615
	Q2	Sum	$2,278,693
Year	Quarter	Month Name	Sales revenue
	Q3	August	$173,756
		July	$525,904
		September	$668,181
	Q3	Sum	$1,367,841
Year	Quarter	Month Name	Sales revenue
	Q4	December	$649,350
		November	$484,024
		October	$655,206
	Q4	Sum	$1,788,580
2001		Sum	$8,095,814

6 Save the document in your Favorites folder as **Doc_Formatbreaks**.

Formatting multiple break headers and footers

There are several ways to modify multiple breaks in tables.

In the table shown below, the values for both Year and Quarter appear only once and have been centered over the break. This was achieved using the Center Value Across Break formatting option.

Year	Quarter	Month name	Sales revenue
2001	Q1	January	$1,003,541
		February	$630,073
		March	$1,027,085
	Q2	April	$895,260
		May	$865,615
		June	$517,819
	Q3	July	$525,904
		August	$173,756
		September	$668,181
	Q4	October	$655,206
		November	$484,024
		December	$649,350
2001		Sum:	$8,095,814

Year	Quarter	Month name	Sales revenue
	Q1	January	$1,335,402
		February	$609,013
		March	$1,381,758
	Q2	April	$1,068,309
		May	$1,081,885
		June	$690,457

The table header in this table only appears for each Year, not for each Quarter. This was achieved by turning off the break header for Quarter and turning on the break header for Year.

The sum on Sales Revenue displays only the yearly revenue totals, not the quarterly totals. This was achieved by turning off the break footer for Quarter while retaining the break footer for Year.

In the following procedures, you will format the breaks in your table so that it appears like the table above.

To center the value across the break

1 Continue working with the same document.

2 Select the **Year** column in the table.

3 Click the **Properties** tab.

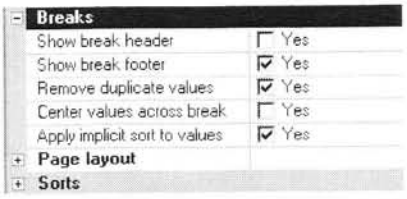
The Properties tab appears in the left pane.

The Properties tab allows you to format a number of elements related to the part of the document that is selected. In this case, because you have selected a data column, you can format the size of the column, the text displayed in the column, the background of the column, and other standard formatting elements.

You can also format the way any breaks and sorts applied to the column are displayed.

4 Click + next to the **Breaks** section to expand the folder and view the options available to you for formatting the break applied to the Year column.

The Breaks folder expands in the Properties tab.



5 Select the **Yes** check box next to the **Center values across break** option.

6 Click the **Quarter** column in the table.

7 In the Properties tab, select the **Center values across break** option again.

The table now appears like this (truncated in this image):

Year	Quarter	Month Name	Sales revenue
2001	Q1	February	$630,073
		January	$1,003,541
		March	$1,027,085
	Q1	Sum	$2,660,700
Year	Quarter	Month Name	Sales revenue
2001	Q2	April	$895,260
		June	$517,819
		May	$865,615
	Q2	Sum	$2,278,693
Year	Quarter	Month Name	Sales revenue
		August	$173,756

To remove break headers and footers

1 Select the **Year** column in the table.

2 Check that both the **Yes** check boxes next to the **Show break header** and the **Show break footer** options are selected.

3 Select the **Quarter** column.

4 **Clear** both of the **Yes** check boxes (so that the options are *not* selected) next to the **Show break header** and **Show break footer** options for the Quarter column so that neither the header or footer rows are displayed for the Quarter break.

5 Click anywhere outside of the table.
The table now appears like this:

Year	Quarter	Month Name	Sales revenue
2001	Q1	February	$630,073
		January	$1,003,541
		March	$1,027,085
	Q2	April	$895,260
		June	$517,819
		May	$865,615
	Q3	August	$173,756
		July	$525,904
		September	$668,181
	Q4	December	$649,350
		November	$484,024
		October	$655,206
2001		Sum	$8,095,814

Year	Quarter	Month Name	Sales revenue
	Q1	February	$609,013
		January	$1,335,402
		March	$1,381,758
	Q2	April	$1,068,309
		June	$690,457

As we saw previously, the Month Name column currently is sorted in the default sort order, that is, in ascending order based on values returned by the measure object included in the query. In this example, the values in the Sales revenue have been sorted accordingly, with months earning the lowest revenue appearing first in the table, per quarter.

If you tried to change the sort order by applying a standard sort on the Month Name column, you could only switch from ascending order to descending order: the rows would be sorted to display the highest earning month in the quarter first, instead of the lowest.

To change the order of the rows so that the months are deposed not based on revenue earned but based on their order during the year, you need to create a custom sort. You will learn to create a custom sort later in this lesson.

6 Save the document.

About formatting breaks

Some important things to remember when using breaks are:

- Place all needed breaks onto the table before placing any calculations. If you place the calculations first and apply a break on an object, the calculation context does not recognize that it should re-calculate to the new break level (that is, create sub-totals). Place the breaks first to ensure that the new calculation context is in place before the calculation takes place.
- If you place an automatic calculation, for example a Sum or a Count, on a table after you format the breaks, you lose all formatting done on the break footers. The default action for these calculations is to calculate on all break levels present, regardless of how they have been formatted.

Prioritizing multiple breaks

When you have multiple breaks in a document, you can change the default order so that one break is displayed as a higher priority than the other.

To set the priority of multiple breaks in a table

1 Continue working with the same document.

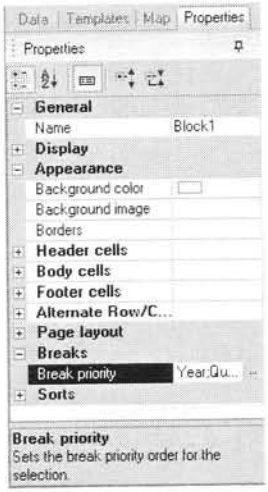
2 Position your cursor over the table until a blue border appears around the table.

3 Right-click the **blue border** and select the **Edit Format** option from the drop-down menu.

The Properties tab appears in the left panel.

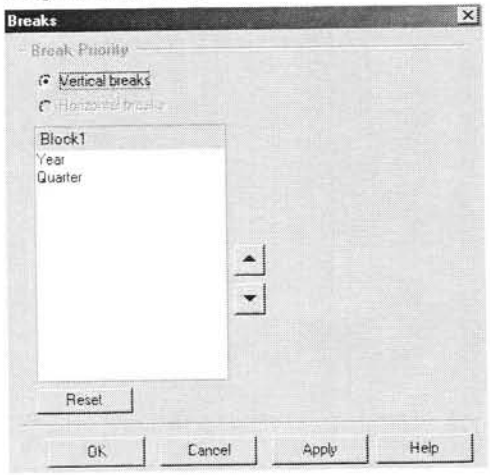
4 Click + next to the **Breaks** folder to see the options available at the table level for formatting breaks.

The Breaks folder expands.



5 Click the **...** (three dots) button to the right of the **Breaks priority** option.

The Breaks dialog box appears.



You can see that currently, the breaks priority is Year followed by Quarter.

6 Use the **arrow** buttons to set **Quarter** at a higher priority than Year.

7 Click **OK**.

The table now appears like this:

Year	Quarter	Month Name	Sales revenue
2001	Q1	February	$630,073
		January	$1,003,541
		March	$1,027,085
2001		Sum:	$2,660,700
Year	**Quarter**	**Month Name**	**Sales revenue**
2002	Q1	February	$609,013
		January	$1,335,402
		March	$1,381,758
2002		Sum:	$3,326,172
Year	**Quarter**	**Month Name**	**Sales revenue**
2003	Q1	February	$863,452
		January	$1,501,367
		March	$1,378,170
2003		Sum:	$3,742,989
Year	**Quarter**	**Month Name**	**Sales revenue**
2001	Q2	April	$895,260
		June	$517,819
		May	$865,615

The data is presented with all first quarters during the three year period at the top of the table, followed by all Q2 values. The sum is calculated at the quarter level, and a total of all first quarter revenue is displayed as well.

8 Return the breaks priority to the original order and save the document.

Using breaks to format crosstabs

Since crosstabs contain two levels of dimension objects (object values are located both in the columns and in the rows of the block), several unique formatting issues may occur. The proper use of breaks on the rows and columns of the crosstab allows you to resolve these issues.

When two or more measure objects are used in a crosstab, an additional header is needed to display the object names.

To use breaks with additional header rows

1 Create a new Web Intelligence document using the eFashion universe.

2 Build a query using the **State**, **Year**, **Sales revenue** and **Margin** objects.

3 Click **Run Query**.

4 Position your cursor on the **table border** until a blue border appears around the table.

5 Right-click the **blue border** and select the **Turn To** option that appears on the drop-down menu.

6 In the Tables tab in the Turn To dialog box, select the **Crosstab** table format.

7 Check that the values returned by **Year** object are positioned as column headers in the crosstab, and the values returned by the **State** object are positioned as row headers.
If this is not the case, swap the headers using the drag-and-drop technique.

The crosstab now appears like this:

	2001		2002		2003	
California	$1,704,211	$774,893	$2,782,680	$1,076,528	$2,992,679	$1,121,489
Colorado	$448,302	$203,701	$768,390	$294,483	$843,584	$309,966
DC	$693,211	$310,356	$1,215,158	$457,231	$1,053,581	$385,415
Florida	$405,985	$192,479	$661,250	$266,670	$811,924	$318,132
Illinois	$737,914	$348,750	$1,150,659	$465,478	$1,134,085	$439,865
Massachusetts	$238,819	$111,453	$157,719	$63,657	$887,169	$336,574
New York	$1,667,696	$779,301	$2,763,503	$1,104,278	$3,151,022	$1,189,166
Texas	$2,199,677	$1,011,038	$3,732,889	$1,459,562	$4,185,098	$1,566,478

Since no secondary header was added to indicate the column names for the measure objects in the body of the crosstab (only the Year header is present), you must add one, so that you can distinguish between the Sales revenue values and the Margin values.

8 Right-click any **Year** column-header cell.
Or, click the drop-down arrow next to the **Insert Row Above** button on the **Reporting** toolbar.

9 Select the **Insert row below** option from the drop-down menu.

10 Click any **cell** in the first data column (*not* the column header).
Notice that all three of the columns are highlighted.

11 Click the **Properties** tab and verify in the **General, Text zone** which measure object is projecting data in this column.
In this case, it should display =[Sales revenue], which indicates the values in these columns is sales revenue data.

12 Click the new **second column header** of one of the columns you highlighted.

13 In the **General, Text** zone of the Properties tab, type the header label: **Sales Revenue**.

Note: Remember to press the Return key when you have typed the text.

14 Click the second column header for the other columns and type the header label: **Margin**.

Note: Another way to edit the header text is to click the row/column header, display the Formula toolbar and type the text in the Formula bar.

The crosstab now appears like this:

	2001		2002		2003	
	Sales Revenue	Margin	Sales Revenue	Margin	Sales Revenue	Margin
alifornia	$1,704,211	$774,893	$2,782,680	$1,076,528	$2,992,679	$1,121,489
olorado	$448,302	$203,701	$768,390	$294,483	$843,584	$309,966
C	$693,211	$310,356	$1,215,158	$457,231	$1,053,581	$385,415
lorida	$405,985	$192,479	$661,250	$266,670	$811,924	$318,132
linois	$737,914	$348,750	$1,150,659	$465,478	$1,134,085	$439,865
assachusetts	$238,819	$111,453	$157,719	$63,657	$887,169	$336,574
ew York	$1,667,696	$779,301	$2,763,503	$1,104,278	$3,151,022	$1,189,166
exas	$2,199,677	$1,011,038	$3,732,889	$1,459,562	$4,185,098	$1,566,478

Notice that the values for the Year object are not centered over the new column headers for Sales revenue and Margin. This is a problem inherent in crosstab formatting. You can place a formatted break on the Year object to work around this problem.

To apply a break on a column header in a crosstab

1 Click in the **Year** row to highlight it.

2 Click **Insert/Remove Break** on the Reporting toolbar.
The crosstab now appears like this:

	2001		2001	2002		2002
	Sales Revenue	Margin		Sales Revenue	Margin	
California	$1,704,211	$774,893		$2,782,680	$1,076,528	
Colorado	$448,302	$203,701		$768,390	$294,483	
DC	$693,211	$310,356		$1,215,158	$457,231	
Florida	$405,985	$192,479		$661,250	$266,670	
Illinois	$737,914	$348,750		$1,150,659	$465,478	
Massachusetts	$238,819	$111,453		$157,719	$63,657	
New York	$1,667,696	$779,301		$2,763,503	$1,104,278	
Texas	$2,199,677	$1,011,038		$3,732,889	$1,459,562	

3 In the **Properties** tab, expand the **Breaks** folder.

4 Select the **Yes** check box next to the **Center values across break** option.

Now that the values for Year are centered over the Sales revenue and margin columns, the multiple crosstabs created by the Break can be reformatted to look like a single crosstab.

Note: Creating a break and centering the value across the break is only necessary because Year is the main object across the row of the crosstab. If the two cells were normal cells, you could use the Merge Cells option (Properties tab, Text format folder) and center the value across the selected cells.

5 To remove both the extra column and the extra white spaces in between each crosstab (these were created when the break was placed on Year), **clear** the **Yes** check box next to the **Show Break Footer** option (not selected).

The crosstab now appears like this:

	2001		2002		2003	
	Sales Revenue	Margin	Sales Revenue	Margin	Sales Revenue	Margin
California	$1,704,211	$774,893	$2,782,680	$1,076,528	$2,992,679	$1,121,489
Colorado	$448,302	$203,701	$768,390	$294,483	$843,584	$309,966
DC	$693,211	$310,356	$1,215,158	$457,231	$1,053,581	$385,415
Florida	$405,985	$192,479	$661,250	$266,670	$811,924	$318,132
Illinois	$737,914	$348,750	$1,150,659	$465,478	$1,134,085	$439,865
Massachusetts	$238,819	$111,453	$157,719	$63,657	$887,169	$336,574
New York	$1,667,696	$779,301	$2,763,503	$1,104,278	$3,151,022	$1,189,166
Texas	$2,199,677	$1,011,038	$3,732,889	$1,459,562	$4,185,098	$1,566,478

6 Save the document in your Favorites folder as **Doc_Breakscrosstabs**.

Practice

Activity: Formatting breaks

Objective

- Format a table using breaks.

Instructions

1 Use the eFashion universe to create the report shown below:

Lines	Category	SKU desc	Sold at (unit price)	Discount
Outerwear	Day wear	Blazer	$129.48	$-111,967
		Shetland Jacket	$169.70	$1,664
	Hats,gloves,scarves	Flower Stem Patterned Silk Scarf	$91.95	$42,829
	Night wear	Twill Dressing-gown	$219.73	$41,716
Lines	**Category**	**SKU desc**	**Sold at (unit price)**	**Discount**
Overcoats	Dry wear	Fake Fur Overcoat	$198.60	$58,948
		Velour Overcoat	$203.47	$148,825
	Wet wear	High Collar Engineers Raincoat	$230.31	$17,370
		Stretch Cotton Raincoat	$211.81	$21,431
		Trenchcoat	$200.99	$58,372
		Sum:	$1,656.04	
			Average:	$31,021

2 Save the document to your Favorites folder as **Act_Formatbreaks**.

Creating custom sorts

Introduction

You can apply sorts to the results displayed in:

- **Section cells** – to organize the order in which sections are displayed in a report
- **Tables** – to organize the order in which results are displayed in a column or row

Sorting sections enables you to organize the section headers logically in a report. For example, if you have created sections on a report for each year, you can apply a descending sort so that the sections are organized with the most recent year as the first section and the earliest year at the end of the report.

You apply sorts to any dimensions, measures, or details displayed on a table. Sorting dimensions and details helps you organize results chronologically, while sorting measures helps you see highest or lowest results at a glance.

This unit describes how to create a custom sort. Custom sorts allow you to define your own order for the data displayed in a report, other than the default orders that are available with standard sorts.

After completing this unit, you will be able to:

- Explain the default sort orders available in Web Intelligence
- Create a custom sort
- Apply a custom sort to a column of data
- Delete a custom sort

About sorts in Web Intelligence

Sorts in Web Intelligence allow you to use the following orders:

Sort order	Description
Default	This is sometimes referred to as the "natural" order. Depending on the type of data in the column or row, the results are sorted as follows: • ascending numeric order for numeric data • ascending chronological order for date • alphabetical order for alphanumeric data
Ascending	When selected, results are arranged in ascending order: The smallest value at the top of the column moving to the highest value at the bottom. For example: 100, 200, 300 or California, Colorado, Florida.

Sort order	Description
Descending	When selected, results are arranged in descending order: The highest value at the top of the column moving to the smallest value at the bottom. For example: 300, 200, 100 or Florida, Colorado, California.
Custom	You define your own sort order.

How is data sorted when you apply a break?

When you insert a break on a dimension, the values for the dimension are automatically sorted in ascending order. If the values are numeric, the lowest value appears in the first row of the table, the highest in the last row. If the values are alphabetical characters, then the values are sorted in alphabetical order from top to bottom. You can change this sort order at any time.

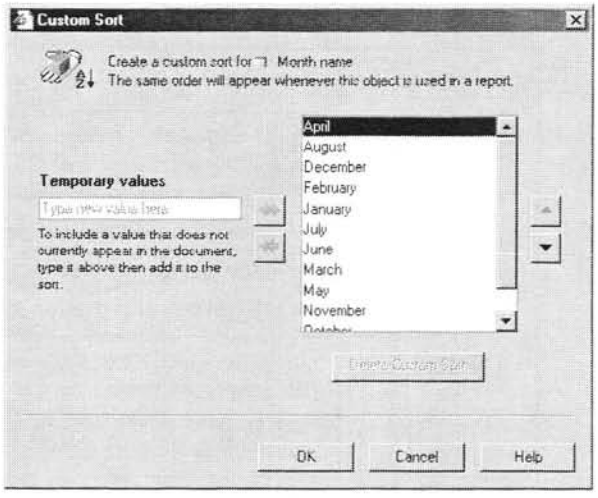
Creating a custom sort

To create a custom sort

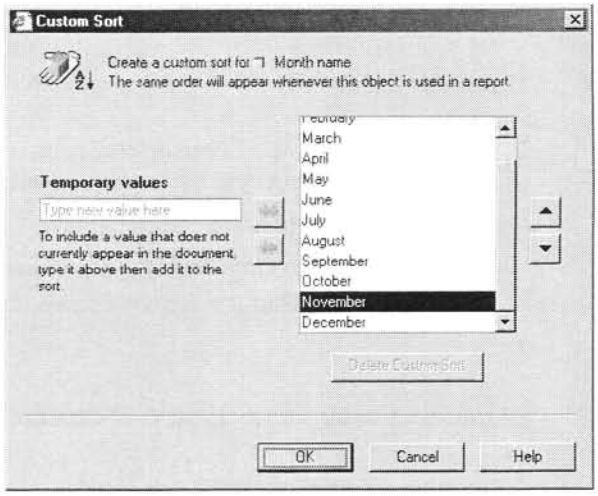
1 Open the **Doc_Formatbreaks** document that you saved to your Favorites folder earlier in this lesson.

2 Right-click the **Month Name** column and select **Sort** from the drop-down menu.
Or, click the drop-down arrow next to the **Apply/Remove Sort** button on the **Reporting** toolbar.

3 Select **Custom sort...** from the drop-down menu that appears.
The Custom Sort dialog box appears.



4 Use the **Up** and **Down** arrow buttons to order the months in the standard order so that the Custom Sort dialog box appears like this:



5 Click **OK** to create the custom sort and close the dialog box.
The table now appears with the data sorted in the order of months in the year:

Year	Quarter	Month name	Sales revenue
2001	Q1	January	$1,003,541
		February	$630,073
		March	$1,027,085
	Q2	April	$895,260
		May	$865,615
		June	$517,819
	Q3	July	$525,904
		August	$173,756
		September	$668,181
	Q4	October	$655,206
		November	$484,024
		December	$649,350
2001		Sum	$8,095,814

Year	Quarter	Month name	Sales revenue
	Q1	January	$1,335,402
		February	$609,013
		March	$1,381,758
	Q2	April	$1,068,309
		May	$1,081,885
		June	$690,457

Deleting a custom sort

To delete a custom sort, select the Delete Custom Sort option in the Custom Sort dialog box.

Displaying document data in free-standing cells

Introduction

This unit describes how you can display certain data available in the document as a free-standing cell in the document reports.

After completing this unit, you will be able to:

- Display the user's response to a prompt in a title cell
- Display the date that the document was last refreshed

Displaying the user's response to a prompt

You want to capture the user's response to a prompt and display the updated value in the title every time the document is opened or refreshed.

To create the document with a prompted query filter

1 Create a new Web Intelligence document using the **eFashion** universe.

2 Build a query using the **Year**, **Quarter** and **Margin** objects.

3 Create a **prompted query filter** that requires the user to select a state every time the report is opened or refreshed.

4 Click **Run Query**.

5 When you are prompted, select **Illinois**.

6 Apply a break on the **Year** column.

7 Apply a **sum** on the **Sales revenue** column.
The report appears like this:

Report Title

Year	Quarter	Margin
2001	Q1	$115,882
	Q2	$114,847
	Q3	$48,520
	Q4	$69,501
2001	Sum:	$348,750

Year	Quarter	Margin
2002	Q1	$128,878
	Q2	$100,760
	Q3	$91,794
	Q4	$144,046
2002	Sum:	$465,478

Year	Quarter	Margin
2003	Q1	$93,109
	Q2	$146,466

You are going to create a formula to capture your response to the prompt. To create the formula, you will use the UserResponse() function, but you will also need to use the DataProvider() function and any of the objects used in the query.

To show prompt input with the UserResponse() function

1 Click the **Report Title** cell to highlight it.

2 Click the **Show/Hide Formula Toolbar** button on the **Reporting** toolbar.
The Formula toolbar appears and displays the text that appears in the title cell currently: Report Title.

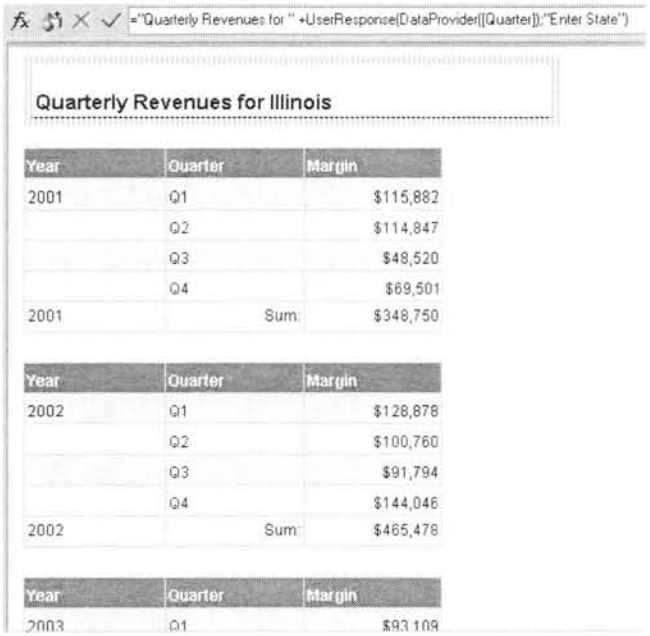
3 Click the **Formula Editor** button on the **Formula** toolbar.
The Formula Editor appears.

4 Type or use the **Data**, **Functions** and **Operators** tab to help you enter the following formula:

```
="Quarterly Revenues for "
+UserResponse(DataProvider([Quarter]);"Enter State")
```

5 Click the **Validate** button to check your syntax.

6 Click **OK** to close the Formula Editor.
The title cell has been updated with the text you entered in the formula, plus the state that you selected when you last ran the report, **Illinois**.



Year	Quarter	Margin
2001	Q1	$115,882
	Q2	$114,847
	Q3	$48,520
	Q4	$69,501
2001	Sum:	$348,750

Year	Quarter	Margin
2002	Q1	$128,878
	Q2	$100,760
	Q3	$91,794
	Q4	$144,046
2002	Sum:	$465,478

Year	Quarter	Margin
2003	Q1	$93,109

7 Click **Refresh Data**.

8 When you are prompted, select **New York**.
The table is updated to show data concerning New York and the title cell is updated as well.

9 Save the document to your Favorites folder as **Doc_Displayprompt**.

Displaying the last refresh date

You want to display the date and time the document was last refreshed in a free-standing cell so that the data will be updated every time the document is refreshed.

To display the last refresh date in a cell

1. Continue working with the **Doc_Displayprompt** document.
2. Click the **Templates** tab in the left panel.
3. Click + next to the Free-Standing Cells folder to expand the folder.
4. Click + next to the Formula and Text Cells folder to expand it.
5. Drag the **Last Refresh Date** template from the Templates tab to the blank space to the right of the existing table.
6. Right-click the **cell** and select **Format Number** from the drop-down menu to change the format of the date and time.
7. Expand the cell or reduce the font size as necessary to view all the text.
8. Insert a **free-standing cell** and type text to identify the Last Refresh Date information.
9. Format the text in the same way you did the Last Refresh Date cell.

The report now appears like this:

Quarterly Revenues for New York

Last time document was refreshed:

Friday, January 13, 2006

Year	Quarter	Margin
2001	Q1	$248,404
	Q2	$230,808
	Q3	$106,053
	Q4	$194,036
2001	Sum	$779,301

Year	Quarter	Margin
2002	Q1	$261,295
	Q2	$281,708
	Q3	$183,699
	Q4	$377,577
2002	Sum	$1,104,278

Year	Quarter	Margin
2003	Q1	$276,708

Displaying data restricted by a filter or ranking

Introduction

This unit describes how you can display certain data that is available in the data provider but does not appear in the table because a report filter or ranking function has been applied to hide some of the values returned by the query.

After completing this unit, you will be able to:

- Use the NoFilter() function to override a report filter or ranking

Overriding a report filter

At times you might need to display a calculation that includes both the data that is shown in the table and data that is present in the data provider, but not currently shown in the table. The only way to achieve this is to force Web Intelligence to ignore a report filter or ranking that has been applied to an object or variable in the data provider.

NoFilter function

Use the NoFilter() function to override a report filter or ranking. NoFilter() includes all of the values for a variable in the data provider in the calculation, even if the data is not displayed in the table or block.

Syntax:

```
NoFilter(AggregateFunction(measure))
```

Scenario

Create a report that displays the top three stores per year. Section the report by year. Calculate the revenue of the top three stores, the revenue of the stores that did not make the top three, and the sum of all the stores combined.

To use the NoFilter() function

1. Create a new Web Intelligence document using the **eFashion** universe.
2. Build a query using the **Year, Store name** and **Sales revenue** objects.
3. Click **Run Query**.
4. Right-click the **Year** column and select **Set as Section** from the drop-down menu.
5. Select the **Sales revenue** column, and then click the **Apply/Remove Ranking** button on the **Reporting** toolbar.

6 In the Rank dialog box, accept the default values to list only the **top three stores** based on **Sales revenue** and click **OK**.

7 Apply a **Sum** to the **Sales revenue** column.
The table should now look like this:

2001

Store name	Sales revenue
e-Fashion New York Magnolia	$1,023,061
e-Fashion Los Angeles	$982,637
e-Fashion Chicago 33rd	$737,914
Sum:	$2,743,612

2002

Store name	Sales revenue
e-Fashion New York Magnolia	$1,687,359
e-Fashion Los Angeles	$1,581,616

8 Right-click the 2001 Sales revenue **footer row** and select **Insert row below** from the drop-down menu.

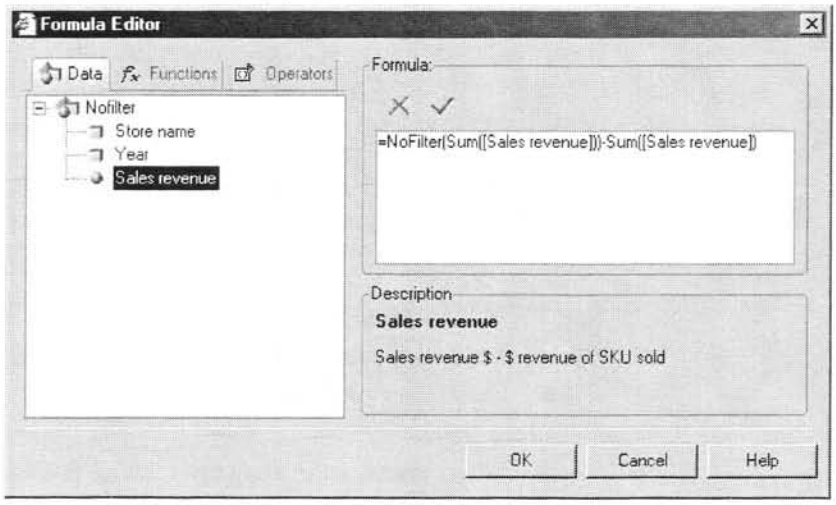
9 Click the new **cell** below the original sum to select it.

10 Click the **Show/Hide Formula Toolbar** button on the **Reporting** toolbar.

11 Type directly in the **Formula** bar, or open the **Formula Editor**, and create a formula that will calculate the sum of Sales revenue for the stores that are **not** part of the top three.

```
=NoFilter(Sum([Sales revenue]))-Sum([Sales revenue])
```

The Formula Editor appears like this:



By dissecting this formula, you can see how the formula will return the desired result.

Formula component:	Syntax and description:
`=NoFilter(Sum([Sales revenue]))`	`NoFilter(Aggregate([Measure]))` This returns the entire sum of all the stores and ignores rank.
`-Sum([Sales revenue])`	This part of the formula returns the sum of the stores in the current block (with the rank). It does not include the stores that are missing.

12 Click **Validate** to check your syntax, and then click **OK** to close the Formula Editor.

At this point, your report should look like this example:

2001

Store name	Sales revenue
e-Fashion New York Magnolia	$1,023,061
e-Fashion Los Angeles	$982,637
e-Fashion Chicago 33rd	$737,914
Sum:	$2,743,612
	5,352,202

2002

Store name	Sales revenue
e-Fashion New York Magnolia	$1,687,359
e-Fashion Los Angeles	$1,581,616
e-Fashion Washington Tolbooth	$1,215,158
Sum:	$4,484,133
	8,748,112.9

2003

13 Edit the text in the cells to the left of the calculated sums, so that the data is immediately meaningful.

2001

Store name	Sales revenue
e-Fashion New York Magnolia	$1,023,061
e-Fashion Los Angeles	$982,637
e-Fashion Chicago 33rd	$737,914
Sum of Top Three:	$2,743,612
Sum of Other Stores:	5,352,202

2002

Store name	Sales revenue
e-Fashion New York Magnolia	$1,687,359
e-Fashion Los Angeles	$1,581,616
e-Fashion Washington Tolbooth	$1,215,158
Sum of Top Three:	$4,484,133
Sum of Other Stores:	8,748,112.9

2003

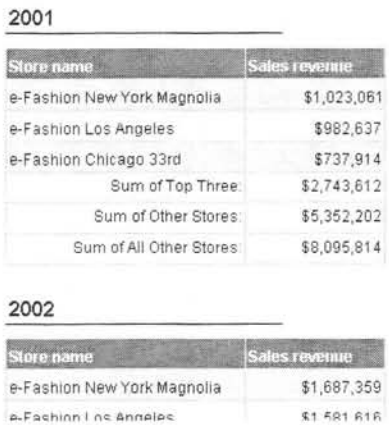
The next and final step is to calculate the sum of all the stores.

14 Insert another **row** at the bottom of the table.

15 Highlight the **cell** below the other sum cells and using the Formula toolbar or the Formula Editor, type this formula:

```
=NoFilter(Sum([Sales revenue]))
```

The finished report should look like this example:



2001

Store name	Sales revenue
e-Fashion New York Magnolia	$1,023,061
e-Fashion Los Angeles	$982,637
e-Fashion Chicago 33rd	$737,914
Sum of Top Three:	$2,743,612
Sum of Other Stores:	$5,352,202
Sum of All Other Stores:	$8,095,814

2002

Store name	Sales revenue
e-Fashion New York Magnolia	$1,687,359
e-Fashion Los Angeles	$1,581,616

Note: The results shown in this example are truncated.

The formula calculates the following data:

=NoFilter(Sum([Sales revenue])) - Sum([Sales revenue])	Result
$8,095,814 - $2,743,612	= $5,352,202

16 Save the document in your Favorites folder as **Doc_NoFilter**.

Practice

Activity: Displaying document information in a report

Objectives

- Build a formula using the UserResponse() function to capture the user's response to a prompt and display it in the title cell.
- Display the date and time the document was last refreshed.

Instructions

1 Use the eFashion universe to create the report shown below.

2 Build the report so that it prompts the user to select a year and a product line.

3 Format the title cell so that it displays the chosen value when the document is refreshed.
The report should look like this:

2001 Annual report for Leather

Store name	Sales revenue	Quantity sold	Margin
e-Fashion Austin	$2,713	15	$1,301
e-Fashion Boston Newbury	$811	5	$272
e-Fashion Chicago 33rd	$2,215	12	$980
e-Fashion Colorado Springs	$3,903	26	$1,579
e-Fashion Dallas	$2,135	11	$1,113
e-Fashion Houston	$1,185	7	$488
e-Fashion Houston Leighton	$2,052	10	$1,177
e-Fashion Los Angeles	$24,031	163	$9,234
e-Fashion Miami Sundance	$1,596	8	$848
e-Fashion New York 5th	$2,837	14	$1,501
e-Fashion New York Magnolia	$5,579	30	$2,530
e-Fashion San Francisco	$2,881	16	$1,258
e-Fashion Washington Tolbooth	$1,359	7	$706

Date was last refreshed on:1/13/06 4:35 PM

4 Save the document to your Favorites folder as **Act_Documentdisplay**.

Lesson summary

Review

Quiz: Advanced Reporting Techniques

1 When you apply a standard calculation such as a Sum or an Average on a column in a table with a break, where is the calculation inserted?

   1 In a data cell
   2 In the break footer
   3 In the section
   4 In the break header

2 In the Java Report Panel, how do you format a break?

3 How do you set the break priority when there are multiple breaks applied to a table?

4 What function can you use to capture the response to a prompted query filter?

5 What function can you use to override a report filter or a ranking?

Summary

After completing this lesson, you are now able to:

- Manipulate break headers and footers to change the display of the data
- Prioritize multiple breaks in a table
- Use breaks to format crosstabs
- Explain the default sort orders available in Web Intelligence
- Create a custom sort
- Apply a custom sort to a column of data
- Delete a custom sort
- Display the user's response to a prompt in a title cell
- Display the date that the document was last refreshed
- Use the NoFilter() function to override a report filter or ranking

Lesson 5
Calculation Contexts

This lesson provides information on how Web Intelligence performs calculations in reports. By default, Web Intelligence determines the result of a measure when it is projected in the report based on the dimension(s) in the part of the report where the measure is inserted (for example, in columns of a table). These dimensions make up what is called the calculation context of the measure object or variable.

Understanding how contexts behave by default is key to understanding how to manipulate the context so that, in certain circumstances, the report displays the data you are interested in.

This lesson demonstrates how to use various operators and keywords to define and redefine contexts in your calculations.

In this lesson you will learn about:

- Understanding calculation contexts
- Redefining calculation contexts

Duration: 1 hour 30 minutes

Understanding calculation contexts

Introduction

This unit introduces key concepts of calculation contexts in Web Intelligence. Calculation context is the way that Web Intelligence dynamically calculates values projected in a report by measure objects or variables.

In a Web Intelligence document, measures are calculated dynamically based on the dimensions with which they appear. For this reason, it is important to understand that Web Intelligence, by default, performs calculations at the row level.

After completing this unit, you will be able to:

- Explain how Web Intelligence calculates data dynamically
- Explain the impact of input and output contexts on how data is calculated
- Explain how you can use extended syntax to change the default calculation context

Dynamic calculations

By default, Web Intelligence dynamically calculates and displays measures based on the dimension that appears in the block with the measure.

State	Year	Sales revenue
California	2001	$1,704,211
California	2002	$2,782,680
California	2003	$2,992,679
Colorado	2001	$448,302
Colorado	2002	$768,390
Colorado	2003	$843,584
DC	2001	$693,211
DC	2002	$1,215,158
DC	2003	$1,053,581
Florida	2001	$405,985
Florida	2002	$661,250
Florida	2003	$811,924
Illinois	2001	$738,224
Illinois	2002	$1,150,659
Illinois	2003	$1,134,085
Massachusetts	2001	$238,819
Massachusetts	2002	$157,719
Massachusetts	2003	$887,169
New York	2001	$1,667,696
New York	2002	$2,763,503
New York	2003	$3,151,022
Texas	2001	$2,199,677
Texas	2002	$3,732,689
Texas	2003	$4,185,098

State	Sales revenue
California	$7,479,569
Colorado	$2,060,275
DC	$2,961,950
Florida	$1,879,159
Illinois	$3,022,968
Massachusetts	$1,283,707
New York	$7,582,221
Texas	$10,117,664

In the first table above, Sales revenue is aggregated in the context of State and Year. When you remove a dimension from the table, as in the second table above, Web Intelligence automatically recalculates the sales revenue data according to the new context (State, in this case).

In this example, sales revenue has been calculated at the row level. This row level calculation is the default calculation context for any measure object or variable placed in a column.

Although this default calculation context is usually sufficient, there are times when you need the measure to calculate at a level other than the default context.

It is possible to change the default context of a measure by modifying the syntax of the variable to include a specific input and/or output context for the calculation.

Input and output contexts

Input and output contexts must be added to a calculation if you want the context of the calculation to be something other than the default context.

- Input context consists of any dimension objects that need to be included directly IN the calculation itself.
- Output context consists of one or more dimension objects that determine where the calculation is placed in the report, or in other words, the level where the calculation is to be turned OUT in the report.
  In fact, the output context determines at what aggregation level the calculation is displayed (for example, a master variable in a section).

In the simple calculation shown here:

```
= Sum([Sales revenue])
```

there are no dimension objects in the calculation itself, so the input context is the sum of the values returned by the measure object Sales revenue. This calculation does not specify an output context, so Web Intelligence assumes the default context and uses the dimensions with which the measure appears.

State	Sales revenue
California	$7,479,569
Colorado	$2,060,275
DC	$2,961,950
Florida	$1,879,159
Illinois	$3,022,968
Massachusetts	$1,283,707
New York	$7,582,221
Texas	$10,117,664
Sum:	$36,387,512

As no output context is specified for the Sum calculation, it assumes the context of State.

The syntax for defining the input and output contexts is:

```
AggregateFx([Measure]input_context) output_context
```

Understanding when to redefine the context

The following example demonstrates how to extend the syntax of a formula in order to redefine the output context that will be used to calculate the measure.

To redefine the output context using extended syntax

1. Create a new Web Intelligence document using the **eFashion** universe.
2. In the Java Report Panel, build a query by dragging or double-clicking the **State**, **Year** and **Sales revenue** objects to move them into the Result Objects pane.
3. Click **Run Query**.
4. In Edit Report view, select the **State** column to highlight it.
5. Click the **Insert/Remove Break** button on the **Reporting** toolbar to break the table into separate blocks.
6. Click the **Sales revenue** column and click the **Insert Sum** button on the **Reporting** toolbar.
7. Verify that the Sales revenue column is still selected, and then click the **Apply/Remove Sort** button on the **Reporting** toolbar and select **Descending**.

   The report now appears as shown below:

State	Year	Sales revenue
California	2003	$2,992,679
	2002	$2,782,680
	2001	$1,704,211
California	Sum:	$7,479,569

State	Year	Sales revenue
Colorado	2003	$843,584
	2002	$768,390
	2001	$448,302
Colorado	Sum:	$2,060,275

State	Year	Sales revenue
DC	2002	$1,215,158

The Sales revenue measure is currently calculated to show each state's total sales revenue per year. You can easily see each state's best year amount as you view the rows in the block. The break footer displays the total revenue earned for all three years.

8 Create a new table in this report by dragging and dropping the **State** and **Sales revenue** objects just to the right of this block.

Note: To select more than one object at a time, click the first object, hold down the Ctrl key and click another object. Continue holding down the Ctrl key and drag both objects at the same time to a position next to the existing block.

The report now appears as shown below:

State	Year	Sales revenue
California	2003	$2,992,679
	2002	$2,782,680
	2001	$1,704,211
California	Sum:	$7,479,569

State	Sales revenue
California	$7,479,569
Colorado	$2,060,275
DC	$2,961,950
Florida	$1,879,159
Illinois	$3,022,968
Massachusetts	$1,283,707
New York	$7,582,221
Texas	$10,117,664

State	Year	Sales revenue
Colorado	2003	$843,584
	2002	$768,390
	2001	$448,302
Colorado	Sum:	$2,060,275

State	Year	Sales revenue
DC	2002	$1,215,158
	2003	$1,053,581
	2001	$693,211

In this new table, you can see each state's overall sales revenue, without either the revenue per year or the state's best year amount. The sales revenue data has been aggregated to the state level.

9 Create a new variable called **State's best year amount**.
Use the following formula in the variable definition:

```
=max([Sales revenue])
```

Note: Qualify the variable as a measure and be sure to validate your syntax before closing the Variable Editor.

10 Drag and drop the new variable so that the data is projected in a new column to the right of the Sales revenue, in the second block.

The report now appears like this:

State	Year	Sales revenue
California	2003	$2,992,679
	2002	$2,782,680
	2001	$1,704,211
California	Sum:	$7,479,569

State	Year	Sales revenue
Colorado	2003	$843,584
	2002	$768,390
	2001	$448,302
Colorado	Sum:	$2,060,275

State	Year	Sales revenue
DC	2002	$1,215,158
	2003	$1,053,581
	2001	$693,211

State	Sales revenue	State's best year amount
California	$7,479,569	$7,479,569
Colorado	$2,060,275	$2,060,275
DC	$2,961,950	$2,961,950
Florida	$1,879,159	$1,879,159
Illinois	$3,022,968	$3,022,968
Massachusetts	$1,283,707	$1,283,707
New York	$7,582,221	$7,582,221
Texas	$10,117,664	$10,117,664

If you compare the state's best year revenue shown in the first block with the data calculated by the variable you just created, you can see that the default behavior of the calculation, at the row level, is obviously not the correct context.

By default, Web Intelligence does not know that the Year object needs to be included in the context of the calculation - it is simply using State as the input context in order to calculate the values, and the result is the exact same calculation that the Sales revenue object projects.

Now you are going to extend the syntax in order to specify the correct input and output context so that Web Intelligence knows which context to use to calculate the data correctly.

11 Modify the variable definition to specify the input and output context, using this formula:

```
=max([Sales revenue] In([Year] ; [State])) In ([State])
```

Compare the total sales revenue for all three years shown in the first block, with the data in the Sales revenue column in the second block.

Web Intelligence now calculates the data correctly, because it calculates the maximum amount of Sales revenue for each State in any one Year.

State	Year	Sales revenue
California	2003	$2,992,679
	2002	$2,782,680
	2001	$1,704,211
California	Sum:	$7,479,569

State	Year	Sales revenue
Colorado	2003	$843,584
	2002	$768,390
	2001	$448,302
Colorado	Sum:	$2,060,275

State	Year	Sales revenue
DC	2002	$1,215,158
	2003	$1,053,581
	2001	$693,211

State	Sales revenue	State's best year amount
California	$7,479,569	$2,992,679
Colorado	$2,060,275	$843,584
DC	$2,961,950	$1,215,158
Florida	$1,879,159	$811,924
Illinois	$3,022,968	$1,150,659
Massachusetts	$1,283,707	$887,169
New York	$7,582,221	$3,151,022
Texas	$10,117,664	$4,185,098

12 Save the document in your Favorites folder as **Doc_Bestyear**.

Components of the input and output contexts

In order to understand input and output contexts, let's review the following components of the calculation:

```
=max([Sales revenue] In([Year] ; [State])) In ([State])
```

Component	Description
`In([Year] ; [State])`	This component is considered the **input** context since the variable needs to consider both the State object and the Year object (which is not in the table) when making the calculation. Notice that the entire input context appears in the same parentheses as the measure object. **Note:** Web Intelligence uses In Body if there are no dimensions in the input context by default.
`In ([State])`	This component is considered the **output** context since it is necessary for the calculation to aggregate at both the State and Year levels (outlined in the input context), but display at the State level only. The output context is always placed outside the parentheses containing the measure object.

Note: This example is given for demonstration purposes. It is not always necessary to define the output context, since Web Intelligence assumes this context based on the placement of the formula in the report. As a rule of thumb, context should be left as the default (undefined), unless it must be defined in order to achieve a certain result. This allows variables to be reused in other parts of the report.

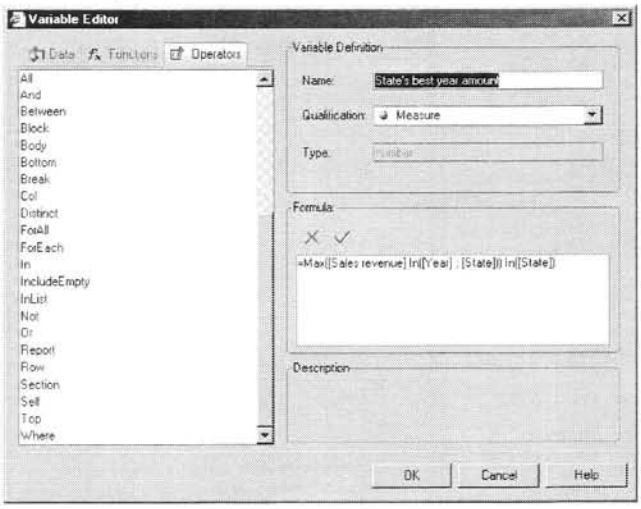
The operator **In** is a generic operator that allows you to define the parameters of either input or output contexts.

About the extended syntax operators and keywords

The extended syntax that is available to redefine the calculation context of measures includes:

- Extended syntax context operators, including In, ForEach, ForAll
- Extended syntax keywords, including Report, Section, Break, Block and Body. These keywords are used in formulas in conjunction with the In context operator.

The operators and keywords you can use to define input and output contexts are located in the Operators tab in the Variable Editor:



Redefining calculation contexts

Introduction

This unit provides examples of reasons why you might choose to redefine the input or output contexts of measures in your report.

These examples show how to use both extended syntax context operators and keywords.

After completing this unit, you will be able to:

- Use the In context operator to redefine calculation context
- Use the In context operator with Where to redefine calculation context
- Use the ForEach context operator to redefine calculation context
- Use the ForAll context operator to redefine calculation context
- Use the Block keyword with In to redefine calculation context

Using extended syntax context operators

The context operators that are available to redefine the calculation context of measures include:

Operator	Definition
In	Used to specify dimensions explicitly. Also used with extended syntax keywords.
Where	Specifies limiting conditions on the data
ForEach	Adds dimensions to the context
ForAll	Removes dimensions from the context

The ForAll and ForEach operators are useful when you have a default context with many dimensions. It is often easier to "add" or "subtract" from the context using ForAll and ForEach than it is to specify the list explicitly using In.

In context operator

The In context operator is used to specify dimensions explicitly in a context.

Syntax:

```
In([dimension] ; [dimension])
```

Scenario

You have a report showing Year and Sales Revenue. The Quarter object is also available in the data provider but you do not include this dimension in the block. Instead, you want to include an additional column to show the maximum revenue by quarter in each year.

To use the In context operator

1. Create a new Web Intelligence document using the **eFashion** universe.
2. In the Java Report Panel, add the **Year**, **Quarter**, **State**, and **Sales revenue** objects to the query.
3. Click **Run Query**.
4. In Edit Report view, remove the **Quarter** and the **State** objects from the table.
   The table should look like this:

Year	Sales revenue
2001	$8,096,124
2002	$13,232,246
2003	$15,059,143

5. Hold down the **Ctrl** key and drag the table to the right to make a **duplicate copy**.
6. In the copied table, drag the **Quarter** object and insert it in a column between the **Year** and **Sales revenue** columns.
7. Apply a break on the **Year** column.
8. Click the **Sales revenue** column to highlight it.

9 Click the drop-down arrow next to the **Insert Sum** button on the Report toolbar and select **Max**.

The report should look like this:

Year	Sales revenue
2001	$8,096,124
2002	$13,232,246
2003	$15,059,143

Year	Quarter	Sales revenue
2001	Q1	$2,660,700
	Q2	$2,279,003
	Q3	$1,367,841
	Q4	$1,788,580
2001	Max:	$2,660,700

Year	Quarter	Sales revenue
2002	Q1	$3,326,172
	Q2	$2,840,651
	Q3	$2,879,303
	Q4	$4,186,120
2002	Max:	$4,186,120

Year	Quarter	Sales revenue
2003	Q1	$3,742,989

In the second table, because Quarter is in the block, the default calculation context automatically displays the maximum quarterly revenue in the footer.

You want to display the same values in the first table, without having to include the Quarter object in the table.

10 Create a Max Quarterly Revenue variable using this formula:

```
=Max([Sales revenue] In([Year] ; [Quarter])) In([Year])
```

11 Drag the new variable to the right of the Sales revenue column in the original table.

The original table looks like this:

Year	Sales revenue	Max Quarterly Revenue
2001	$8,096,124	$2,660,700
2002	$13,232,246	$4,186,120
2003	$15,059,143	$4,006,718

This formula tells Web Intelligence to calculate the maximum sales revenue for each (Year, Quarter) combination, and then output this figure by year.

Note: Because the default output context of the block is Year, you do not need to specify the output context explicitly in this formula.

12 Save the document to your Favorites folder as **Doc_Contextoperators.**

In context operator with Where

Two of the most widely-used operators used to specify both input and output contexts are In and Where. Although each operator denotes a different type of calculation environment, the two may be used together in a single variable.

The In operator specifies all parameters (dimension objects) that should be included in the context. When using multiple dimension objects, the dimensions should be listed in order of granularity and should be separated with ";".

The Where operator instructs the variable to calculate only where certain values are true.

Scenario

Produce a report that calculates both the highest Sales revenue for all states as well as specific information for California.

To use the In and Where context operators

1 Continue working with the **Doc_Contextoperators** document.

2 Right-click the **Report 1** report tab and select **Insert Report** from the drop-down menu.

3 In the new report, press the **Ctrl** key and drag the **Year** and **Sales revenue** objects to create a new vertical table.
The table should look like the following:

Year	Sales revenue
2001	$8,096,124
2002	$13,232,246
2003	$15,059,143

4 Create a **Sales revenue By Year for California** variable with following input context:

```
=[Sales revenue] Where ([State]="California")
```

This context uses the Where operator to isolate the California state in the calculation.

5 Replace the **Sales revenue** object in the table with the **new variable**.

Note: Use the Properties tab to format the column in the table:

- Set the text in the column header cell to Wrap
- Display the data as $ amounts.

Your report should now look like this example:

Year	Sales revenue By Year for California
2001	$1,704,210.80
2002	$2,782,679.50
2003	$2,992,679.00

6 Create the **Sales revenue for all States** calculation by dragging and dropping the **Sales revenue** object from the Data tab and dropping it outside the table anywhere in the report space.

7 Insert a **free-standing cell** to the left of this calculation and enter text that identifies the calculation.

8 Create the **Highest Sale Revenue for any one State** variable using this syntax:

```
=Max([Sales revenue] In([State]))
```

This context uses the In operator to specify input context. This assures that the calculation only addresses the State values when locating a maximum amount.

There is no output context defined, so the calculation uses the default output context of the entire report, because it was placed at the report level rather than in a table.

9 Drag the new variable in the report space and insert a free-standing cell with text that identifies the calculation.

10 Create the **Highest Annual Revenue for California** variable using the following syntax:

```
==Max([Sales revenue] In([State];[Year]) Where([State]
="California"))
```

This variable definition uses both the In and Where operators to achieve the correct calculation. Both operators were used in defining the input context.

- The In operator forces the =Max function to look in the State values first and then the Year values within each State in order to locate the maximum value.
- The Where operator assures that the =Max calculation is only valid where the State is California.

11 Create the **Sales revenue for California for 2003** variable using this syntax:

```
=[Sales revenue] Where([State]="California" And [Year]
="2003")
```

Like the Sales revenue By Year for California object, this calculation uses the Where operator in the input context to ensure that the calculation is valid for only those values denoted by the Where operators.

It is not necessary to list the components of the calculation in order of granularity when using the Where operator. You must list the components when using the In operator.

12 Insert this variable in the report and insert another free-standing cell with text to identify the calculation.

Your completed report should look like this example:

Sales revenue for all states:	**$36,387,512**
Highest sales revenue for any one State:	**$10,117,664**
Highest annual revenue for California:	**2,992,679**
Sales revenue for California for 2004:	**2,992,679**

Year	Sales revenue By Year for California
2001	$1,704,210.80
2002	$2,782,679.50
2003	$2,992,679.00

13 Save the document.

ForEach context operator

The ForEach context operator adds dimensions to a context.

Syntax:

```
ForEach([dimension])
```

Scenario

Create a report that calculates the maximum revenue by year and quarter in a table, when the Quarter object is not shown in the table.

To use the ForEach context operator

1 Continue working with the **Doc_Contextoperators** document.

1 Right-click the **Report 2** report tab and select **Insert Report** from the drop-down menu.

2 In the new report, press the **Ctrl** key and drag the **Year** and **Sales revenue** objects to create a new vertical table.

3 Create a **Max Quarterly Revenue** variable with the following input context:

```
=Max([Sales revenue] ForEach([Quarter])) In([Year])
```

4 Insert the new variable in the table.
The table should look like the following:

Year	Sales revenue	Max Quarterly Revenue
2001	$8,096,124	$2,660,700
2002	$13,232,246	$4,186,120
2003	$15,059,143	$4,006,718

The Year dimension is the default input context in the block. By using the ForEach operator, you add the Quarter dimension to the context, giving an input context of ([Year];[Quarter]).

5 Save the document.

ForAll context operator

The ForAll context operator removes dimensions to a context.

Syntax:

```
ForAll([dimension])
```

Scenario

You have a report showing sales revenue per year and per quarter. You want to add a column that shows the total revenue in each year.

To use the ForAll context operator

1 Continue using the **Doc_Contextoperators** document.

1 Right-click the **Report 3** report tab and select **Insert Report** from the drop-down menu.

2 In the new report, press the **Ctrl** key and drag the **Year, Quarter** and **Sales revenue** objects to create a new vertical table.

3 Click the **Sales revenue** column and apply a **Sum**.
The standard sum function inserts a footer in the table and displays the total revenue for all quarters in the three years, using the default calculation context.

Year	Quarter	Sales revenue
2001	Q1	$2,660,700
2001	Q2	$2,279,003
2001	Q3	$1,367,841
2001	Q4	$1,788,580
2002	Q1	$3,326,172
2002	Q2	$2,840,651
2002	Q3	$2,879,303
2002	Q4	$4,186,120
2003	Q1	$3,742,989
2003	Q2	$4,006,718
2003	Q3	$3,953,395
2003	Q4	$3,356,041
	Sum	$36,387,512

You want to display the total revenue for each year. To total revenues by year, the input context needs to be (Year); by default, it is (Year; Quarter). Therefore, you need to remove Quarter from the input context by specifying ForAll ([Quarter]) in the formula, which looks like this:

```
=Sum([Sales revenue]) ForAll ([Quarter])
```

Note: Note that you can use the In operator to achieve the same thing. Using In, the formula is:

```
=Sum([Sales Revenue] In ([Year]))
```

This version of the formula explicitly specifies Year as the context, rather than removing Quarter to leave Year.

4 Create a **Yearly Total** variable with the following input context:

```
=Sum([Sales revenue]) ForAll ([Quarter])
```

5 Insert the new variable in the table.
The table should look like the following:

Year	Quarter	Sales revenue	Yearly Total
2001	Q1	$2,660,700	$8,096,124
2001	Q2	$2,279,003	$8,096,124
2001	Q3	$1,367,841	$8,096,124
2001	Q4	$1,788,580	$8,096,124
2002	Q1	$3,326,172	$13,232,246
2002	Q2	$2,840,651	$13,232,246
2002	Q3	$2,879,303	$13,232,246
2002	Q4	$4,186,120	$13,232,246
2003	Q1	$3,742,989	$15,059,143
2003	Q2	$4,006,718	$15,059,143
2003	Q3	$3,953,395	$15,059,143
2003	Q4	$3,356,041	$15,059,143
	Sum:	$36,387,512	

Note: Another way to show this calculation would be to apply a break on the Year column and then apply a standard sum. You would then get the yearly total in the break footer, and the overall total in the block footer.

6 Save the document.

Using extended syntax keywords

The extended syntax keywords, Report, Section, Break, Block and Body, are also useful in defining calculation context. These keywords, used in conjunction with the In operator, allow you to change the default context from the row level to another level in the document.

In this example, you will use the In Block keyword to change the calculation context from the row level to the block (or table) level, in order to highlight values with an alerter.

Scenario

Create a report that contains an alerter which highlights any store whose revenue for the year 2003 is lower than the average revenue of all stores in 2003.

To calculate averages in a table and highlight below average results

1. Create a new Web Intelligence document using the eFashion universe.
2. In the Java Report Panel, build a query using the **Store name** and **Sales revenue** objects.
3. Apply a **query filter** to restrict the data retrieved to the year **2003**.
4. Click **Run Query**.
5. In Edit Report view, click the **Sales revenue** column to highlight it.
6. Click the drop-down arrow next to the **Insert Sum** button on the **Reporting** toolbar and select **Average** from the drop-down list. The table appears like this:

Store name	Sales revenue
e-Fashion Austin	$1,135,479
e-Fashion Boston Newbury	$887,169
e-Fashion Chicago 33rd	$1,134,085
e-Fashion Colorado Springs	$843,584
e-Fashion Dallas	$803,421
e-Fashion Houston 5th	$910,451
e-Fashion Houston Leighton	$1,335,747
e-Fashion Los Angeles	$1,656,676
e-Fashion Miami Sundance	$811,924
e-Fashion New York Magnolia	$1,911,434
e-Fashion New York Sundance	$1,239,587
e-Fashion San Francisco	$1,336,003
e-Fashion Washington Tolbooth	$1,053,581
Average:	$1,158,396

7. Click the **footer cell** at the bottom of the Store name column to highlight it, and then click the **Properties** tab.
8. Edit the text **Average** in the footer so that the cell clearly identifies the calculation at the bottom of the table. Type **Average Revenue for All Stores:**.

9 Click the **Average value** in the footer at the bottom of the table to highlight it.

10 Click the **Show/Hide Formula Toolbar** on the **Reporting** toolbar.
Notice the formula that Web Intelligence used to automatically calculate the average:

```
=Average([Sales revenue])
```

This calculation is generic; it has no input or output context specified. It returns a correct average of revenue earned by stores because no other specific context was defined. The calculation used the default context of the table footer, which contains calculations based only on data that resides in the block (the table, in this case).

Note: If a report filter is applied to the table, the default context in the table footer calculates only the values retained by the filter.

11 Build a new variable called **Average** and use the same syntax used by Web Intelligence to calculate the average revenue for all stores.

12 Drag this new variable and position it to the right of the **Sales revenue** column.
The table now appears like this:

Store name	Sales revenue	Average
e-Fashion Austin	$1,135,479	$1,135,479
e-Fashion Boston Newbury	$887,169	$887,169
e-Fashion Chicago 33rd	$1,134,085	$1,134,085
e-Fashion Colorado Springs	$843,584	$843,584
e-Fashion Dallas	$803,421	$803,421
e-Fashion Houston 5th	$910,451	$910,451
e-Fashion Houston Leighton	$1,335,747	$1,335,747
e-Fashion Los Angeles	$1,656,676	$1,656,676
e-Fashion Miami Sundance	$811,924	$811,924
e-Fashion New York Magnolia	$1,911,434	$1,911,434
e-Fashion New York Sundance	$1,239,587	$1,239,587
e-Fashion San Francisco	$1,336,003	$1,336,003
e-Fashion Washington Tolbooth	$1,053,581	$1,053,581
Average Revenue for All Stores:	$1,158,396	

The Average variable appears in the table, but the values are now being calculated at the row level rather than at the whole table level.

Why is this happening? The default calculation context has changed.

When the formula `=Average([Sales revenue])` is placed in:

- the footer: the context is to calculate for the entire table
- the table: the context is to calculate for each individual row

If you try to build an alerter to highlight values in the Sales revenue column that are below the average, the alerter will not work since the Sales revenue values are currently equal to the average values.

To redefine the calculation context using In Block

1 Double-click the **Average** variable in the **Data** tab.

2 In the **Variable Editor** dialog box, modify the **Average** formula by extending the syntax like this:

```
=Average([Sales revenue]) In Block
```

Why use this syntax?

By adding In Block as the output context, you are specifying that the calculation should be displayed at the block (whole table) level, even though the calculation was placed at the row level.

Why is there no input context?

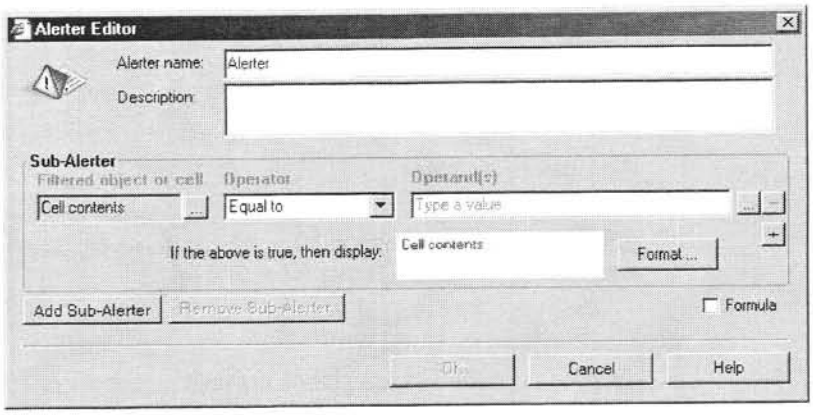
No input context is needed since the calculation needs to find the average based on the entire table. By leaving the input context to assume the default, you assure that no additional objects are considered when the average is calculated.

Now that the Average variable uses the correct formula, you can create the alerter to highlight below average results per store.

To build an alerter to highlight below average results

1 In Edit Report view, click the **Alerters** button on the **Reporting** toolbar. The Alerters dialog box appears.

2 In the Alerters dialog box, click **New**.
The Alerter Editor dialog box appears.



3 In the **Alerter name** field, type **Below Average**.

4 In the **Sub-Alerter** zone, click the square option box just to the right of the **Filtered object or cell** field.
The Available Objects and Variables dialog box appears.

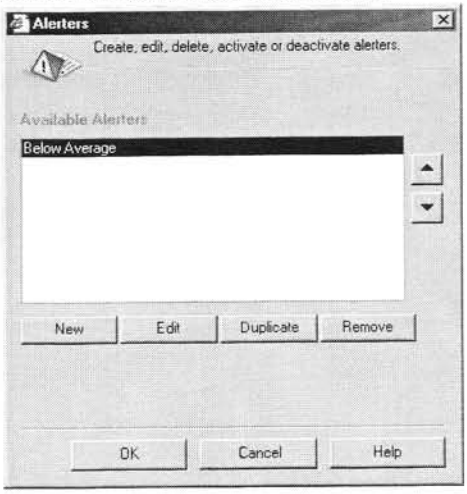
5 Click **Sales revenue** and click **OK**.

6 In the Alerter Editor, click the **Operators** drop down list arrow, then select the **Less than or Equal to** operator.

7 Click the **Expand** box just to the right of the **Operands** field to select an object to compare to.
The Available Objects and Variables dialog box appears.

8 In the Available Objects and Variables dialog box, select the **Average** variable.

9 Click **OK** to create the alerter and close the Alerter Editor.
The Alerters dialog box appears again. Note that the alerter you just created is listed now in the Available Alerters list.

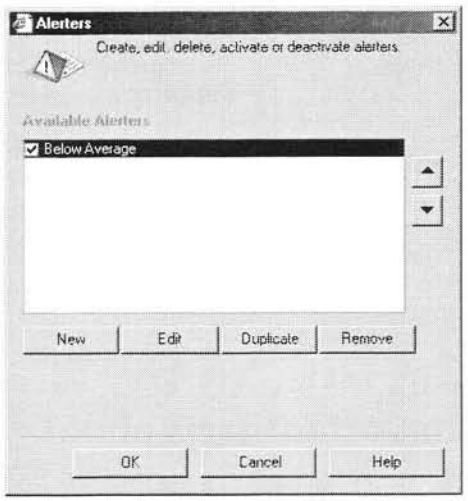


10 Click **OK** to close the Alerters dialog box.
You have finished creating the alerter, but now you must apply it to the appropriate column of data.

11 In Edit Report view, click the **Sales revenue** column in the table to highlight.

12 Click the **Alerters** button on the Reporting toolbar.
The Alerters dialog box appears again.

13 In the Alerters dialog box, verify that the **check box** next to the new alerter is selected.
This option applies the alerter to the column or cell that is highlighted in the block.



14 Click **OK** to close the Alerters dialog box.
The Sales revenue values that are lower than the average revenue for all stores value are highlighted in red in the table.

15 Remove the **Average column** of the table as it is no longer necessary.
The table should appear like this:

Store name	Sales revenue
e-Fashion Austin	$1,135,479
e-Fashion Boston Newbury	$887,169
e-Fashion Chicago 33rd	$1,134,085
e-Fashion Colorado Springs	$843,584
e-Fashion Dallas	$803,421
e-Fashion Houston 5th	$910,451
e-Fashion Houston Leighton	$1,335,747
e-Fashion Los Angeles	$1,656,676
e-Fashion Miami Sundance	$811,924
e-Fashion New York Magnolia	$1,911,434
e-Fashion New York Sundance	$1,239,587
e-Fashion San Francisco	$1,336,003
e-Fashion Washington Tolbooth	$1,053,581
Average Revenue for All Stores	$1,158,396

16 Save the document as **Doc_InBlock**.

More about the extended syntax keywords

The following keywords can be used with the In operator to control how measures are calculated at different levels within the document: Report, Section, Break, Block and Body.

Environment Level	Effect
Report	The projected value of the measure is aggregated for all dimensions contained within the page of the report.
Section	The projected value of the measure is aggregated for all dimensions contained within the section of the report.
Break	The projected value of the measure is aggregated for all dimensions contained within the break of the table.
Block	The projected value of the measure is aggregated across all values for dimensions contained within the block.
Body	The projected value of the measure is aggregated for all dimensions at the level in the report that it is placed.

Important facts about calculation contexts

- If you do not define an input or output context, Web Intelligence assumes the default context of where the calculation has been placed.
- When using variables as the basis of an alerter (as in the Average example just seen), it is necessary to ensure that the calculation has been created as a variable (that is, a formula with a name), and not as a simple formula.

  Note: If the calculation is a formula, show the Formula toolbar and click the Variable Editor button in order to name it as a variable.
- When using the Where operator, it is necessary to place parentheses around the values listed after the Where, as in the following example:

```
=[Sales revenue] Where ([Year] = "2001")
```

Practice

Activity: Calculation contexts

Objective

Use extended syntax to redefine a calculation context of measures in a Web Intelligence document.

Instructions

1 Create a document that shows in a single block:
   - Sales revenue for each eFashion store
   - Total sales revenue for all stores
   - Average revenue for all stores
   - Average revenue for all product lines per store
   - Highlight those stores whose revenue was above average revenue for all stores

2 Change the name of the report to **Revenue Information for all Stores**.

3 Save the document as **Act_Calculationcontext.rep**.
Your document should look like this example:

Revenue Information for All Stores

Store name	Sales revenue	Average Revenue for All Product Lines
e-Fashion Austin	$2,699,673	$2,799,016
e-Fashion Boston Newbury	$1,283,707	$2,799,016
e-Fashion Chicago 33rd	$3,022,658	$2,799,016
e-Fashion Colorado Springs	$2,060,275	$2,799,016
e-Fashion Dallas	$1,970,034	$2,799,016
e-Fashion Houston	$2,303,183	$2,799,016
e-Fashion Houston Leighton	$3,144,774	$2,799,016
e-Fashion Los Angeles	$4,220,929	$2,799,016
e-Fashion Miami Sundance	$1,879,159	$2,799,016
e-Fashion New York 5th	$2,960,366	$2,799,016
e-Fashion New York Magnolia	$4,621,854	$2,799,016
e-Fashion San Francisco	$3,258,641	$2,799,016
e-Fashion Washington Tolbooi	$2,961,950	$2,799,016
Sum:	$36,387,203	
Average:	$2,799,016	

Lesson summary

Review

Quiz: Calculation Contexts

1 What is an input context?

2 What is an output context?

3 Which of the following is an extended syntax context operator?

   1 NoFilter()
   2 ForEach
   3 In Block

4 Which of the following is an extended syntax keyword?

   1 NoFilter()
   2 ForEach
   3 In Section

Summary

After completing this lesson, you are now able to:

- Explain how Web Intelligence calculates data dynamically
- Explain the impact of input and output contexts on how data is calculated
- Explain how you can use extended syntax to change the default calculation context
- Use the In context operator to redefine calculation context
- Use the In context operator with Where to redefine calculation context
- Use the ForEach context operator to redefine calculation context
- Use the ForAll context operator to redefine calculation context
- Use the Block keyword with In to redefine calculation context

Appendix A
Answer Key

This appendix contains the answers to the reviews and/or activities for the applicable lessons.

Lesson 1

Quiz: Advanced Query Techniques

1 What are the three types of query techniques used in this lesson?

Answer:
Combining, Sub-query, Duplicating a query to build another

2 If you were to use the UNION operator to combine queries, what would be the result?

Answer:
Any records would be returned. This is the equivalent of doing an OR between query filters.

3 If you were to use the INTERSECTION operator to combine queries, what would be the result?

Answer:
Only records that matched ALL criteria would be returned. This is the equivalent of doing an AND between query filters.

4 If you were to use the MINUS operator to combine queries, what would be the result?

Answer:
Records that matched one condition but not the other.

5 Give an example of why a query filter won't return data when a combined query using the MINUS does?

Answer:
Web Intelligence will not return data that does not exist using a standard query filter.

6 Can you do a sub-query and return exactly the same results as a combined query?

Answer:
Depending on the query, yes.

7 Why would you choose to do a combined query rather than a sub-query?

Answer:
Combined queries tend to be faster at returning the data.

Lesson 2

Quiz: Character and Date String Functions

1 Name at least three character string functions used in this lesson.
Answer:
Choice of Right(), Substr(), Pos(), Replace(), Length()

2 To find the occurrence of a comma in an object's value, which function would you use?
Answer:
Pos()

3 What is the syntax for the Replace() function?
Answer:
Replace([string] ; "old string" ; "new string")

4 What does *concatenation* mean?
Answer:
To join two or more items together in a string

5 Give an example of something concatenated.
Answer:
Lastname+ ", " + Firstname

6 What happens if you put text and a date together in a variable?
Answer:
You get an error.

7 How do you resolve this?
Answer:
Use the FormatDate() function to convert the date to a character, but also tell it what "picture" to use for the date

8 Can you do calculations on a date?
Answer:
Yes

Lesson 3

Quiz: Using If Logic

1 What is another term for the logic used by the If() function?

Answer:
If...Then...Else logic

2 Which of the following are examples of why you might use the If() function to group values in a variable:

1 To display categories of values

2 To combine values from two different objects of the same type

3 To show values aggregated at higher levels of detail

Answer:
1 and 3

3 True/False. You can use the If() function to define a variable so that a different calculation is used depending on the value retrieved by the object specified in the variable's formula.

Answer:
True

Lesson 4

Quiz: Advanced Reporting Techniques

1 When you apply a standard calculation such as a Sum or an Average on a column in a table with a break, where is the calculation inserted?

   1 In a data cell

   2 In the break footer

   3 In the section

   4 In the break header

   Answer:
   (2)In the break footer

2 In the Java Report Panel, how do you format a break?

   Answer:
   Select the column with the break, go to the Breaks folder in the Properties tab.

3 How do you set the break priority when there are multiple breaks applied to a table?

   Answer:
   Right-click the table, select Edit Format and click the Break Priority setting in the Properties tab.

4 What function can you use to capture the response to a prompted query filter?

   Answer:
   UserResponse().

5 What function can you use to override a report filter or a ranking?

   Answer:
   NoFilter().

Lesson 5

Quiz: Calculation Contexts

1 What is an input context?

Answer:
The input context is the list of dimensions that are included in the calculation itself.

2 What is an output context?

Answer:
The output context is the dimension level in which the calculation is to be displayed in the report. For example, the output context can be the dimension displayed at the row level; the dimension displayed at the break level, and so on.)

3 Which of the following is an extended syntax context operator?

1 NoFilter()

2 ForEach

3 In Block

Answer:
2) ForEach

NoFilter() is a function. In Block is an extended syntax keyword.

4 Which of the following is an extended syntax keyword?

1 NoFilter()

2 ForEach

3 In Section

Answer:
1) In Section

Notes

Training Evaluation

We continuously strive to provide high quality trainers and training guides. To do this, we rely on your feedback. We appreciate you taking the time to complete this evaluation.

Name: _____ **Position:** _____

Company: _____

Email: _____ **Phone:** _____

Course Information

Date: _____ **Course:** _____

Instructor's Name: _____

Location of Training: _____

How would you rate the following?	Excellent	Good	Average	Poor
Did the course content meet your needs?	❑	❑	❑	❑
Were the training materials clear and easy to understand?	❑	❑	❑	❑
Was the Learner's Guide complete and accurate?	❑	❑	❑	❑
Were there enough hands-on activities?	❑	❑	❑	❑
Did the instructor have strong presentation skills?	❑	❑	❑	❑
Did the instructor have a technical understanding of the product?	❑	❑	❑	❑
Was the instructor attentive to the class level, adjusting the lessons accordingly?	❑	❑	❑	❑
Was the training facility properly equipped?	❑	❑	❑	❑
How would you rate the course overall?	❑	❑	❑	❑

What did you enjoy most about the course?

What did you enjoy least about the course?

Comment on this training guide's completeness, accuracy, organization, usability, and readability.

How did you find out about the training class?

What changes, if any, would you like to recommend regarding this session?