

**IBM Cognos Framework Manager:
Design Metadata Models (v10.2.2)**
Student Guide Volume 1
Course Code: B5A52

This material is meant for IBM Academic Initiative use only. NOT FOR RESALE

IBM Cognos Framework Manager: Design Metadata Models (v10.2.2)
B5A52
ERC: 1.0
Published January 2015

All files and material for this course, B5A52 IBM Cognos Framework Manager: Design Metadata Models, are IBM copyright property covered by the following copyright notice.

© Copyright IBM Corp. 2003, 2015

US Government Users Restricted Rights – Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM corp.

IBM, the IBM logo, ibm.com and IBM DB2 are trademarks of International Business Machines Corp., registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available on the Web at "Copyright and trademark information" at www.ibm.com/legal/copytrade.shtml.

Adobe, and the Adobe logo are either registered trademarks or trademarks of Adobe Systems Incorporated in the United States, and/or other countries.

Microsoft, Windows, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.

Java and all Java-based trademarks and logos are trademarks or registered trademarks of Oracle and/or its affiliates.

This material is meant for IBM Academic Initiative use only. NOT FOR RESALE

Contents

PREFACE.....	P-1
CONTENTS.....	P-3
COURSE OVERVIEW	P-9
DOCUMENT CONVENTIONS	P-12
WORKSHOPS	P-13
ADDITIONAL TRAINING RESOURCES.....	P-14
IBM PRODUCT HELP	P-15
OVERVIEW OF IBM COGNOS BI.....	1-1
OBJECTIVES	1-3
IBM SMARTER ANALYTICS	1-4
ANTICIPATE TO SEE, PREDICT, AND SHAPE BUSINESS OUTCOMES	1-6
IBM COGNOS BUSINESS INTELLIGENCE (BI) CAPABILITIES.....	1-8
IBM COGNOS 10 FAMILY.....	1-9
IBM COGNOS BI ENTERPRISE COMPONENTS	1-10
IBM COGNOS BI ARCHITECTURE (HIGH LEVEL).....	1-12
IBM COGNOS BI SECURITY	1-13
IBM COGNOS BI GROUPS AND ROLES.....	1-14
PACKAGE/DATA SOURCE RELATIONSHIP	1-15
DEMO 1: EXPLORE IBM COGNOS BI.....	1-16
EXTEND IBM COGNOS BI ENTERPRISE	1-36
SUMMARY	1-37
IDENTIFYING COMMON DATA STRUCTURES	2-1
OBJECTIVES	2-3
EXAMINE THE ROLE OF AN IBM COGNOS METADATA MODEL	2-4
GOALS OF A DATA MODELER.....	2-5
DATA SOURCES AND MODEL TYPES	2-6
DIFFERENTIATE DATA ENTITIES	2-7
RELATIONAL MODELS: OPERATIONAL VS REPORTING	2-8
EXAMINING OPERATIONAL DATABASES.....	2-9
EXAMPLE: OPERATIONAL DATABASE QUERY	2-10
EXAMPLE: REPORTING DATABASE QUERY	2-11
MODELING A VIRTUAL STAR SCHEMA	2-12

This material is meant for IBM Academic Initiative use only. NOT FOR RESALE

© 2003, 2015, IBM Corporation

This guide contains proprietary information which is protected by copyright. No part of this document may be photocopied, reproduced, or translated into another language without a legal license agreement from IBM Corporation.

P-3

EXAMINING OPERATIONAL DATA.....	2-13
EXAMINING REPORTING DATA	2-14
EXAMINING FACT TABLES	2-15
EXAMINING DIMENSION TABLES	2-16
DEFINING RELATIONSHIPS	2-17
ISSUES WITH STAR SCHEMAS	2-18
EXAMINING RELATIONSHIPS: CARDINALITY	2-19
OPTIONAL VS MANDATORY CARDINALITY.....	2-20
EXAMINING DATA TRAPS	2-21
EXAMINING DATA TRAPS: CHASM TRAP	2-22
EXAMINING DATA TRAPS: TRANSITIVE RELATIONSHIP	2-23
EXAMINING DATA TRAPS: FAN TRAP	2-24
EXAMINING DATA TRAPS: CONNECTION TRAP	2-25
EXAMINING OLAP DATA STRUCTURES.....	2-26
EXAMINING OLAP: MOLAP VS ROLAP	2-27
IDENTIFY DATA ACCESS STRATEGIES	2-28
SUMMARY	2-29
DEFINING REQUIREMENTS	3-1
OBJECTIVES	3-3
MODELING RECOMMENDATIONS	3-4
ANALYZING DATA REQUIREMENTS	3-8
GATHERING REQUIREMENTS	3-9
OVERVIEW OF THE SAMPLE OUTDOORS COMPANY	3-10
IDENTIFYING REQUIREMENTS	3-11
DEMO 1: EXPLORE THE DATA SOURCES	3-14
BEST PRACTICE: MODELING IN STAGES	3-17
MODELING IN LAYERS	3-18
MODELING IN LAYERS: NO MIDDLE LAYER	3-19
MODELING IN LAYERS: BUSINESS LOGIC VIEW	3-20
MODELING IN LAYERS: CONSOLIDATION VIEW	3-21
FOUNDATION OBJECTS VIEW.....	3-22
SUMMARY	3-23

CREATING A BASELINE PROJECT.....	4-1
OBJECTIVES	4-3
WHAT IS FRAMEWORK MANAGER?	4-4
HOW DOES FRAMEWORK MANAGER CONNECT TO IBM COGNOS BI?	4-5
FRAMEWORK MANAGER QUERY MODES.....	4-6
FRAMEWORK MANAGER MODEL TYPES.....	4-7
FRAMEWORK MANAGER PROJECTS	4-8
DEFINING METADATA ELEMENTS	4-9
WHAT DO REPORT AUTHORS VIEW?	4-11
DEMO 1: EXAMINE THE FINAL PROJECT	4-12
FRAMEWORK MANAGER WORKFLOW	4-24
IMPORTING METADATA	4-25
GENERATING RELATIONSHIP CRITERIA	4-26
REQUIREMENTS REVIEW.....	4-27
DEMO 2: CREATE A BASELINE PROJECT	4-28
DEMO 3: PUBLISH A PACKAGE.....	4-37
IMPORTING ADDITIONAL METADATA.....	4-41
REQUIREMENTS: STAFF BY LOCATION	4-42
DEMO 4: EXTEND THE MODEL TO ADD STAFF LOCATION METADATA	4-43
REQUIREMENTS: RETAILER BY LOCATION	4-46
SUMMARY	4-47
WORKSHOP 1: ENHANCE THE MODEL TO ADD RETAILER DATA.....	4-48
PREPARING REUSABLE METADATA	5-1
OBJECTIVES	5-3
FRAMEWORK MANAGER WORKFLOW	5-4
VERIFYING RELATIONSHIPS	5-5
EXAMPLE: MANDATORY CARDINALITY	5-6
EXAMPLE OPTIONAL CARDINALITY	5-7
DEMO 1: VERIFY RELATIONSHIPS.....	5-8
VERIFYING QUERY ITEM PROPERTIES	5-17
DEMO 2: VERIFY AND MODIFY QUERY ITEM PROPERTIES	5-18
SUMMARY	5-27
WORKSHOP 1: VERIFY AND MODIFY QUERY ITEM PROPERTIES.....	5-28

MODELING FOR PREDICTABLE RESULTS: IDENTIFY REPORTING ISSUES.....

6-1

OBJECTIVES	6-3
FRAMEWORK MANAGER WORKFLOW	6-4
SINGLE-FACT QUERIES	6-5
MULTI-FACT QUERY ISSUES	6-6
STITCH QUERIES	6-9
CARDINALITY IN IBM COGNOS	6-10
WHEN DOES IBM COGNOS GENERATE STITCH QUERIES?	6-13
WHEN YOU MAY NOT WANT A STITCH QUERY	6-14
IDENTIFY REPORTING TRAPS	6-15
REFERENCING TWO FACTS WITH NO DIMENSION CONTEXT.....	6-16
DEMO 1: REFERENCE TWO FACTS WITH NO DIMENSION CONTEXT	6-18
GENERATING UNWANTED QUERY SPLITS.....	6-25
DEMO 2: IDENTIFY UNWANTED QUERY SPLITS	6-26
AMBIGUOUS JOINS	6-30
DEMO 3: IDENTIFY AMBIGUOUS JOINS	6-31
USING TOOLS TO ANALYZE THE MODEL	6-34
DEMO 4: USE TOOLS TO ANALYZE THE MODEL	6-35
AMBIGUOUS QUERY SUBJECTS.....	6-39
SUMMARY	6-42
WORKSHOP 1: MODEL IN FREEHAND TO IDENTIFY QUERY USAGE	6-43

MODELING FOR PREDICTABLE RESULTS: VIRTUAL STAR SCHEMAS.....

7-1

OBJECTIVES	7-3
FRAMEWORK MANAGER WORKFLOW	7-4
MODIFYING QUERY SUBJECTS AND RELATIONSHIPS.....	7-5
OPERATIONAL VS. REPORTING DATABASES	7-6
MODELING VIRTUAL STAR SCHEMAS.....	7-7
WHY MODEL AS A STAR SCHEMA?.....	7-8
BENEFITS OF USING MODEL QUERY SUBJECTS	7-9
USING MODEL QUERY SUBJECTS TO OVERRIDE RELATIONSHIPS	7-10
DEMO 1: USE MODEL QUERY SUBJECTS TO RESOLVE AMBIGUOUS QUERY PATHS	7-11
USING MODEL QUERY SUBJECTS AS VIEWS	7-18

This material is meant for IBM Academic Initiative use only. NOT FOR RESALE

REQUIREMENTS REVIEW: SALES FACT	7-19
DEMO 2: CREATE A SALES FACT MODEL QUERY SUBJECT	7-20
WORKSHOP 1: CREATE STAFF RELATED MODEL QUERY SUBJECTS	7-27
WORKSHOP 2: MERGE QUERY SUBJECTS TO CONTROL USAGE.....	7-32
IDENTIFYING RECURSIVE RELATIONSHIPS.....	7-35
RESOLVING RECURSIVE RELATIONSHIPS.....	7-36
DEMO 3: RESOLVE A RECURSIVE RELATIONSHIP	7-37
MODIFYING RELATIONSHIPS.....	7-41
DEMO 4: CREATE A COMPLEX RELATIONSHIP EXPRESSION	7-42
SUMMARY	7-47
MODELING FOR PREDICTABLE RESULTS: CONSOLIDATE METADATA	8-1
OBJECTIVES	8-3
FRAMEWORK MANAGER WORKFLOW	8-4
REQUIREMENTS REVIEW: RETURNS.....	8-5
EXAMINE RETURNS DATA	8-6
CREATING VIRTUAL FACTS	8-7
DEMO 1: CREATE A VIRTUAL FACT	8-8
CREATING VIRTUAL DIMENSIONS	8-13
WORKSHOP 1: CREATE A VIRTUAL DIMENSION	8-14
CONSOLIDATING METADATA.....	8-18
CONSOLIDATION EXAMPLE.....	8-19
REQUIREMENTS REVIEW: PRODUCT SALES	8-20
DEMO 2: CREATE THE CONSOLIDATION VIEW.....	8-21
REQUIREMENTS REVIEW: SALES TARGET	8-26
REQUIREMENTS REVIEW: BRANCH BY LOCATION.....	8-27
WORKSHOP 2: CONSOLIDATE AND SIMPLIFY THE MODEL FOR PRESENTATION.....	8-28
SUMMARY	8-35

CALCULATIONS AND FILTERS.....	9-1
OBJECTIVES	9-3
FRAMEWORK MANAGER WORKFLOW	9-4
CREATING CALCULATIONS	9-5
DEMO 1: CREATE EMBEDDED CALCULATIONS	9-6
FILTERING DATA.....	9-10
DEMO 2: CREATE EMBEDDED AND STANDALONE FILTERS	9-11
CUSTOMIZING METADATA FOR RUN TIME	9-16
IBM COGNOS ENVIRONMENT SESSION PARAMETERS	9-17
PARAMETER MAPS	9-18
MACROS.....	9-19
DYNAMICALLY RETRIEVE LANGUAGE COLUMN	9-20
DEMO 3: CALCULATE BASED ON LOCAL LANGUAGE	9-21
FILTERING BY LOCALE.....	9-26
WORKSHOP 1: FILTER LOCALE LANGUAGE	9-27
WORKSHOP 2: CALCULATE BASED ON LOCAL LANGUAGE	9-30
SUMMARY	9-33
IMPLEMENTING A TIME DIMENSION	10-1
OBJECTIVES	10-3
FRAMEWORK MANAGER WORKFLOW	10-4
REPORT WITHOUT A TIME DIMENSION.....	10-5
DEMO 1: REPORT WITHOUT A TIME DIMENSION.....	10-6
IMPLEMENT A TIME DIMENSION	10-8
DEMO 2: IMPLEMENT A TIME DIMENSION	10-9
RESOLVE MULTIPLE AMBIGUOUS JOINS.....	10-16
DEMO 3: RESOLVE MULTIPLE AMBIGUOUS JOINS	10-17
SUMMARY	10-22

This material is meant for IBM Academic Initiative use only. NOT FOR RESALE

Course Overview

Course Overview

IBM Cognos Framework Manager: Design Metadata Models (v10.2.2) is a five-day, instructor-led course that provides participants with introductory to advanced knowledge of metadata modeling concepts, and how to model metadata for predictable reporting and analysis results using Framework Manager. Participants will learn the full scope of the metadata modeling process, from initial project creation, to publishing of metadata to the Web, enabling end users to easily author reports and analyze data.

Intended Audience

This course is recommended for developers who design metadata models for use in IBM Cognos BI.

Topics Covered

Topics covered in this course include:

- Overview of IBM Cognos BI
- Identifying common data structures
- Gathering requirements
- Creating a baseline project
- Preparing reusable metadata
- Modeling for predictable results
- Creating calculations and filters
- Implementing a time dimension
- Specifying determinants
- Creating the presentation view
- Working with different query subject types
- Setting security in Framework Manager

This material is meant for IBM Academic Initiative use only. NOT FOR RESALE

© 2003, 2015, IBM Corporation

This guide contains proprietary information which is protected by copyright. No part of this document may be photocopied, reproduced, or translated into another language without a legal license agreement from IBM Corporation.

- Creating analysis objects
- Managing OLAP data sources
- Advanced generated SQL concepts and complex queries
- Using parameters in Framework Manager
- Modeling maintenance and extensibility
- Optimizing Framework Manager models
- Working in a multi-modeler environment
- Managing packages in Framework Manager

Course Prerequisites

Participants should have:

- Knowledge of common industry standard data structures and design
- Experience with SQL
- Experience gathering requirements and analyzing data

Course Structure

This course has been developed so that some modules build on each other. If you skip a module, you may encounter issues as you try to complete other modules.

Note that the course has been tested in sequential order of the modules. Any deviation from this order may produce different results, including screen captures of directory structures and saved reports.

Course Environment

The environment provided in this course requires the following services to be started before you begin performing demos and workshops:

- Apache Directory Server
- DB2 -DB2COPY 1 - DB2
- DB2 Remote Command Server (DB2COPY1)
- DB2 Governor (DB2COPY1)

This material is meant for IBM Academic Initiative use only. NOT FOR RESALE

- DB2DAS - DB2FAS00
- IBM Cognos
- Windows Activation Technology Services
- World Wide Web Publishing

To review the services, in the System tray of your environment, click the Services icon, and ensure that the above services are running. If you have closed your image and launched it again, it is a best practice to review the status of the services before continuing with your demos and workshops.

If Apache Directory Server has stopped, be sure to stop the IBM Cognos service, start the Apache Directory Server service, and then start the IBM Cognos service once Apache has started successfully. You can start and stop a specific service by double-clicking the service to open the Properties dialog box, and then clicking the Stop or Start buttons.

Document Conventions

Conventions used in this guide follow Microsoft Windows application standards, where applicable. As well, the following conventions are observed:

Bold

Bold style is used in demo and workshop step-by-step solutions to indicate either:

- actionable items

(Point to **Sort**, and then click **Ascending.**)

- text to type or keys to press

(Type **Sales Report**, and then press **Enter.**)

- UI elements that are the focus of attention

(In the **Format** pane, click **Data**)

Italic

Used to reference book titles.

CAPITALIZATION

All file names, table names, column names, and folder names appear in this guide exactly as they appear in the application.

To keep capitalization consistent with this guide, type text exactly as shown.

Workshops

Workshop Format

Workshops are designed to allow you to work according to your own pace. Content contained in a workshop is not fully scripted out to provide an additional challenge. Refer back to demonstrations if you need assistance with a particular task. The workshops are structured as follows:

The Business Question Section

This section presents a business-type question followed by a series of tasks. These tasks provide additional information to help guide you through the workshop. Within each task, there may be numbered questions relating to the task. Complete the tasks by using the skills you learned in the module. If you need more assistance, you can refer to the Task and Results section for more detailed instruction.

The Task and Results Section

This section provides a task based set of instructions that presents the question as a series of numbered tasks to be accomplished. The information in the tasks expands on the business case, providing more details on how to accomplish a task. Screen captures are also provided at the end of some tasks and at the end of the workshop to show the expected results.

Additional Training Resources

Bookmark [Business Analytics Product Training](http://www-01.ibm.com/software/analytics/training-and-certification/) <http://www-01.ibm.com/software/analytics/training-and-certification/> for details on:

- Instructor-led training in a classroom or online
- Self-paced training that fits your needs and schedule
- Comprehensive curricula and training paths that help you identify the courses that are right for you
- IBM Business Analytics Certification program
- Other resources that will enhance your success with IBM Business Analytics Software

This material is meant for IBM Academic Initiative use only. NOT FOR RESALE

P-14

© 2003, 2015, IBM Corporation

This guide contains proprietary information which is protected by copyright. No part of this document may be photocopied, reproduced, or translated into another language without a legal license agreement from IBM Corporation.

IBM Product Help

Help type	When to use	Location
Task-oriented	You are working in the product and you need specific task-oriented help.	<i>IBM Product - Help link</i>
Books for Printing (.pdf)	<p>You want to use search engines to find information. You can then print out selected pages, a section, or the whole book.</p> <p>Use Step-by-Step online books (.pdf) if you want to know how to complete a task but prefer to read about it in a book.</p> <p>The Step-by-Step online books contain the same information as the online help, but the method of presentation is different.</p>	Start/Programs/ <i>IBM Product/Documentation</i>
IBM on the Web	<p>You want to access any of the following:</p> <ul style="list-style-type: none"> • Training and Certification Web site • Online support • IBM Web site 	<ul style="list-style-type: none"> • http://www-01.ibm.com/software/analytics/training-and-certification/ • http://www-947.ibm.com/support/entry/portal/Overview/Software • http://www.ibm.com

This material is meant for IBM Academic Initiative use only. NOT FOR RESALE

© 2003, 2015, IBM Corporation

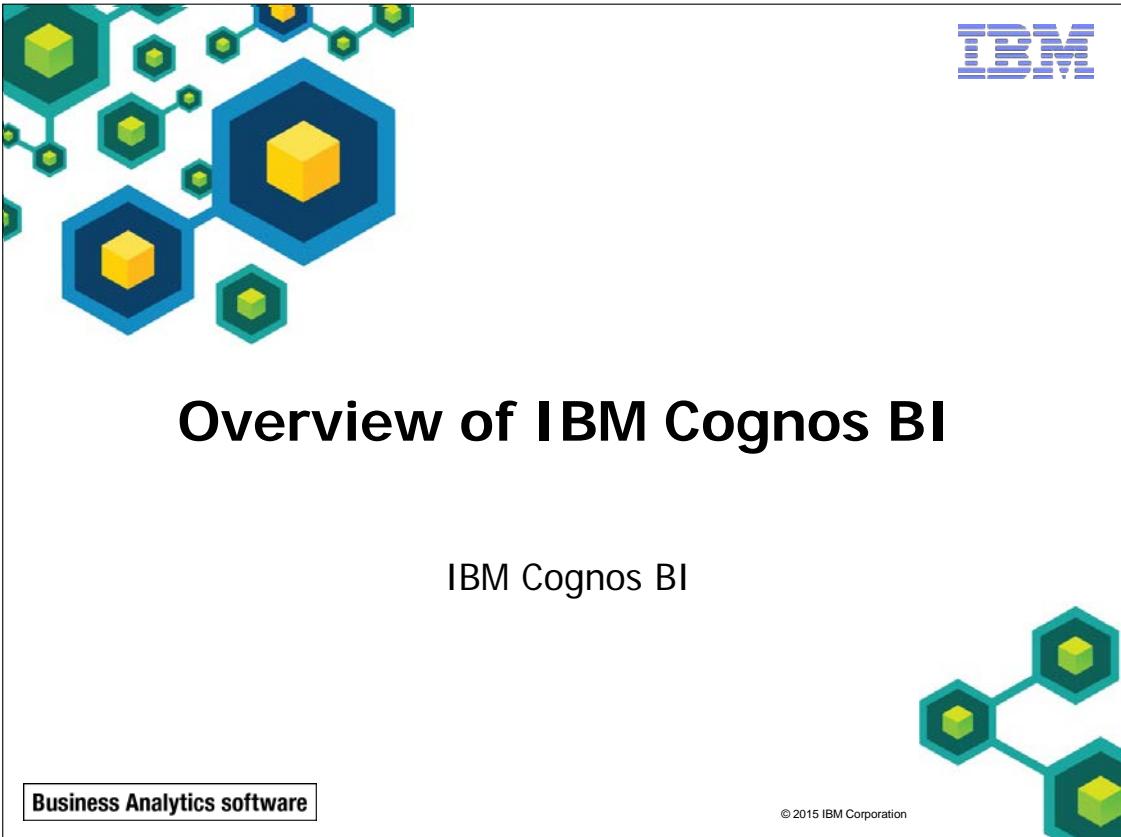
This guide contains proprietary information which is protected by copyright. No part of this document may be photocopied, reproduced, or translated into another language without a legal license agreement from IBM Corporation.

P-15

This material is meant for IBM Academic Initiative use only. NOT FOR RESALE

© 2003, 2015, IBM Corporation

This guide contains proprietary information which is protected by copyright. No part of this document may be photocopied, reproduced, or translated into another language without a legal license agreement from IBM Corporation.



The slide features a white background with a decorative graphic of blue and green hexagons containing yellow cubes in the top left corner. In the top right corner is the blue IBM logo. Below the logo is a large, bold, black title: "Overview of IBM Cognos BI". In the center of the slide is a smaller, bold, black subtitle: "IBM Cognos BI". At the bottom left, there is a black rectangular box containing the white text "Business Analytics software". At the bottom right, there is a small line of fine print: "© 2015 IBM Corporation".

Overview of IBM Cognos BI

IBM Cognos BI

Business Analytics software

© 2015 IBM Corporation

This material is meant for IBM Academic Initiative use only. NOT FOR RESALE

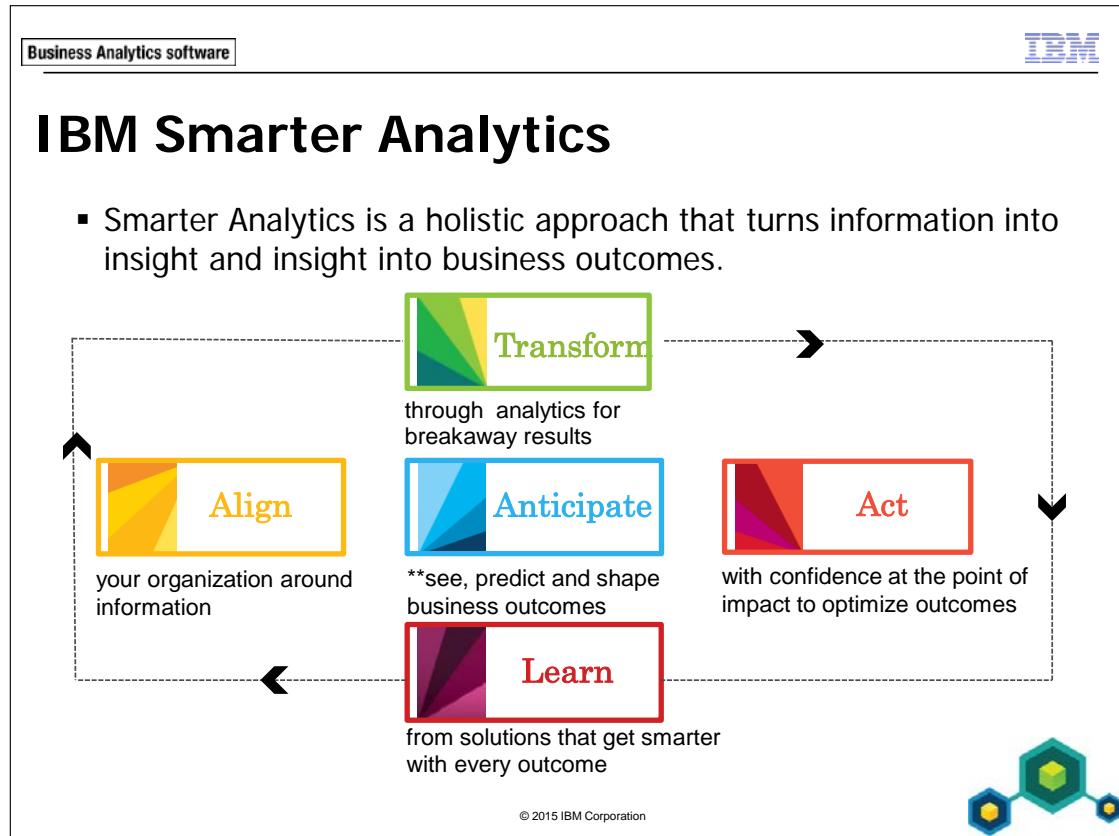
© 2003, 2015, IBM Corporation

This guide contains proprietary information which is protected by copyright. No part of this document may be photocopied, reproduced, or translated into another language without a legal license agreement from IBM Corporation.

Objectives

- At the end of this module, you should be able to:
 - describe IBM Cognos Business Intelligence (BI) and its position within the IBM Smarter Analytics approach and offerings
 - describe the IBM Cognos 10 Family of offerings
 - describe IBM Cognos BI enterprise components
 - describe IBM Cognos architecture at a high level
 - describe IBM Cognos BI security at a high level
 - explain how to extend IBM Cognos BI

© 2015 IBM Corporation

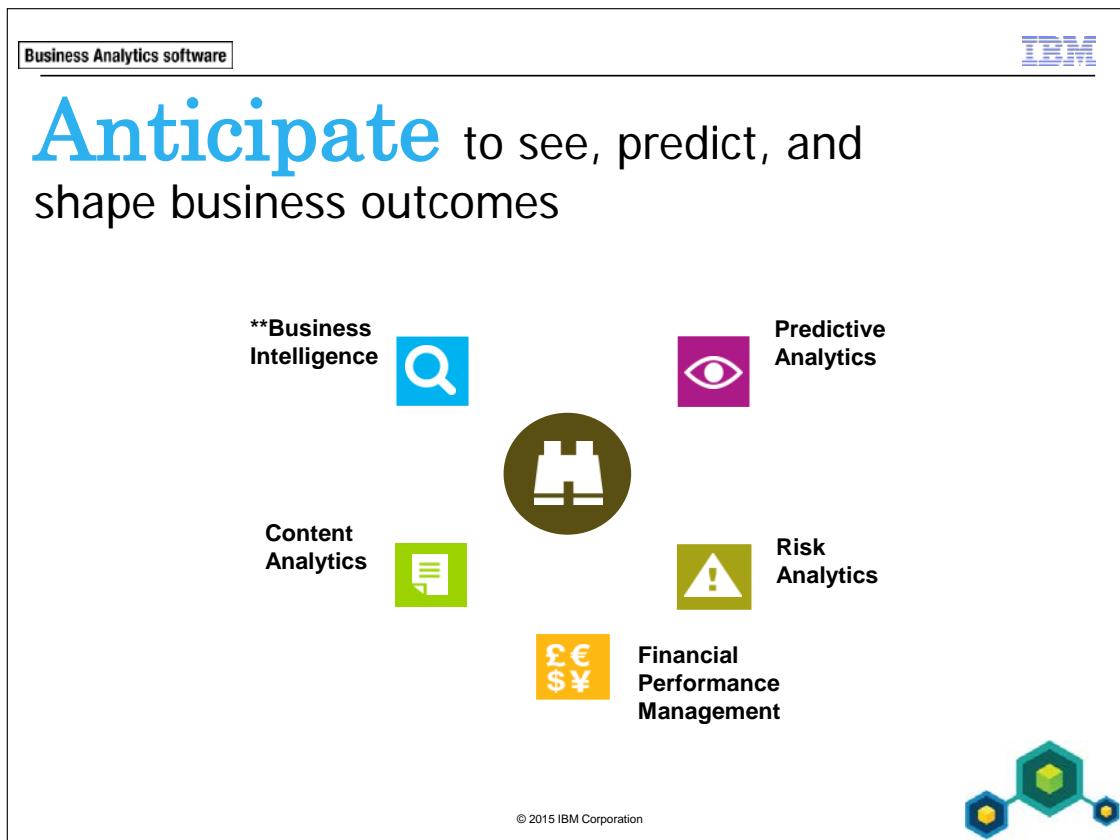


Organizations across industries face tough new challenges created by the information age. A hyper-connected, global community of empowered individuals and consumers is generating an unprecedented amount of big data from billions of diverse sources. Amidst these new complexities, successful organizations are using analytics to acquire, grow and retain customers, transform their financial processes, improve operational efficiency and manage and reduce risk and fraud. Analytics has evolved from a business initiative to a business imperative. Organizations are adopting analytics at a fast rate and those leaders are already transforming entire industries.

From a study conducted by IBM Institute of Business Value and MIT Sloan Management Review, the number of enterprises using analytics to create a competitive advantage jumped almost 60 percent in just one year. Nearly 6 out of 10 organizations now differentiate through analytics. The overall increase in advantage went almost exclusively to organizations who were already experienced users of analytics, so the early adopters are extending their leadership, and those organizations are more than twice as likely to substantially outperform their peers.

This material is meant for IBM Academic Initiative use only. NOT FOR RESALE

The implications of this pattern are clear, and a major transformation is underway. This transformation is fundamentally changing how organizations are structured, how daily operations are managed and where new investments are made to create value. It is being powered by the onset of big data, which in turn is being instrumented and analyzed by new computing systems with deep analytic capabilities. Analytics has grown beyond enterprise data to big, largely unstructured data from billions and billions of diverse sources: . . . there are 200 million tweets sent each day, or roughly 12 TB data. . . every second of high-definition video creates 2,000 times as many bytes as a single page of printed text . . . all told, there are 1.8 trillion gigabytes of information available in today's digital world . . . a truly remarkable figure that continues to grow at an alarming rate. Yet, it is through the complexities caused by big data that we can start to recognize new patterns that we simply couldn't before: 1) insurance companies are identifying fraud patterns by combing different data sources in real time to analyze massive transactional databases 2) financial institutions that are trading based on trending social content 3) energy companies that are analyzing 350 billion meter readings each year to predict power consumption. With every new challenge created by big data comes an equal opportunity; for those who are truly prepared to leverage it; to significantly improve organizational decision making.



Leverage business analytics to deliver actionable insights.

- Spot and analyze trends and anomalies
- Predict potential threats and opportunities
- Plan, budget, and forecast resources
- Assess and manage risk
- Compare "what-if" scenarios
- Measure and monitor business performance
- Automate decisions
- Align strategic and operational decisions

This material is meant for IBM Academic Initiative use only. NOT FOR RESALE

Business Intelligence - capabilities offered by IBM Cognos BI suite of products

Predictive Analytics - capabilities offered by IBM SPSS suite of products

Risk Analytics - capabilities offered by Open Pages and Algorithmics suite of products

Financial Performance Management - capabilities offered by IBM Clarity, IBM Cognos Controller, IBM Cognos Planning, and IBM Cognos TM1

Content Analytics - capabilities offered by IBM Cognos Content Analytics

Business Analytics software

IBM

IBM Cognos Business Intelligence (BI) Capabilities

- IBM Cognos BI provides a range of analytics capabilities so that everyone has the relevant information needed to drive your business forward.

	Reporting		Workspaces		Collaboration
	Analysis		Mobile		Planning and Budgets
	Scorecards		Statistics		Real-time monitoring

© 2015 IBM Corporation

With IBM Cognos BI, users can:

- explore information freely, analyze key facts, collaborate to gain alignment with key stakeholders and make decisions for better business outcomes.
- access reports, analysis, dashboards, scorecards, planning and budgets, real-time information, statistics and manage information for more informed decisions.
- integrate the results of "what-if" analysis modeling and predictive analytics into a unified workspace to view possible future outcomes alongside current and historical data.
- work with business intelligence capabilities for the office and desktop, on mobile devices, online and offline.
- work within a highly scalable and extensible solution that can adapt to the changing needs of IT and the business with flexible deployment options that include the cloud, mainframes and data warehousing appliances.

IBM Cognos 10 Family

- Cognos Insight - Individuals who require personal, desktop analytics
- Cognos Express - Departments, business units or midsize organizations with workgroups who require integrated reporting, analysis and planning
- Cognos Enterprise - Enterprises that require broad analytics capabilities deployed to hundreds or thousands of people

© 2015 IBM Corporation



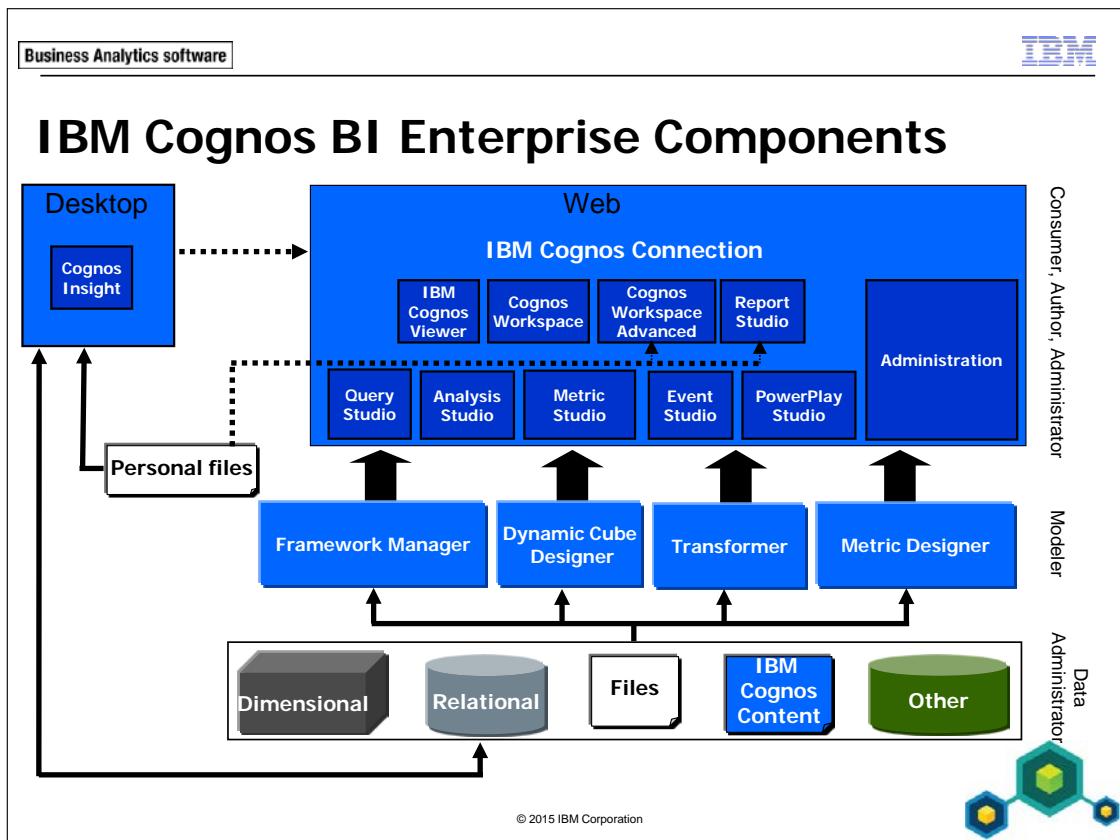
The IBM Cognos 10 family of products is right-sized for your organization and integrated together, and offer solutions that meet your current and future needs, whether you want to deploy on a desktop, a single server, a server farm or all three. You can also start small and grow your solution over time. For example:

- Start small, using Cognos Insight for data discovery and planning. Add a server to share that insight and create additional reports from larger data sets with Cognos Express. Or combine that insight with real-time and corporate information and place insights on scorecards and interact on mobile devices with Cognos Enterprise.

This material is meant for IBM Academic Initiative use only. NOT FOR RESALE

© 2003, 2015, IBM Corporation

This guide contains proprietary information which is protected by copyright. No part of this document may be photocopied, reproduced, or translated into another language without a legal license agreement from IBM Corporation.



IBM Cognos BI capabilities provide reporting, analysis, scorecarding, workspace creation, business event management, and data integration from a wide array of corporate and personal data sources. IBM Cognos BI includes:

- IBM Cognos Connection, which is the Web a portal for BI content presentation, management, and administration.
- Web and desktop reporting and analysis tools to author and analyze corporate data.
- Metadata modeling tools, including Framework Manager, Dynamic Cube Designer, and Transformer.

Use IBM Cognos Viewer to view reports.

Use IBM Cognos Workspace to create personal workspaces.

Use IBM Cognos Workspace Advanced to perform self-service reporting and analyses of data, including external data files.

Use IBM Cognos Insight to perform personal analysis in a desktop environment.

Use Query Studio to perform ad hoc querying and quickly answer a focused question.

Use Analysis Studio to perform analyses of data to discover trends, risks, and opportunities.

Use Report Studio to build sophisticated reports, against multiple data sources, including external data files.

Use Event Studio to create agents which notify users of key operational or performance-related events in their business.

Use PowerPlay Studio to perform multidimensional analysis using IBM Cognos PowerCubes.

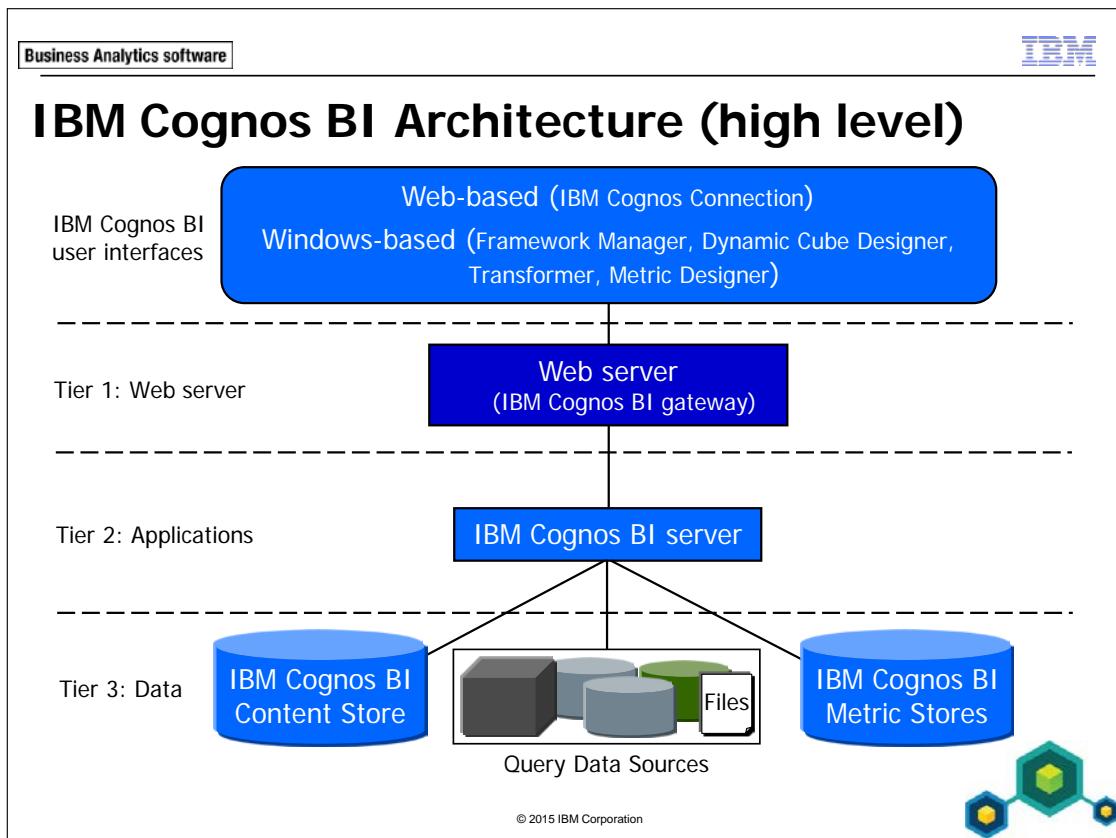
Use Metric Studio to manage performance by monitoring and analyzing metrics.

Use Framework Manager to create basic query packages, relationally based packages, and dimensional analysis packages.

Use Dynamic Cube Designer to create, edit, import, export, and deploy virtual cube models over a relational warehouse schema.

Use Transformer to create PowerCubes for dimensional analysis.

Use Metric Designer to create scorecard applications for use in Metric Studio.



IBM Cognos BI is a Web-based architecture, which is separated into three tiers; Web server, applications, and data.

This architecture is scalable from a software and hardware perspective. For example, you can have several IBM Cognos servers for faster response times and load balancing.

IBM Cognos leverages existing corporate IT resources such as web servers, authentication providers, and application servers, and also supports multiple languages and locales in order to serve a global audience.

IBM Cognos is customizable to adopt your corporate look and feel and can be extended and integrated into other applications through the IBM Cognos SDK.

IBM Cognos BI Security

- The IBM Cognos BI security model combines existing enterprise security solutions with IBM Cognos BI security to achieve:
 - Authentication - Who are you?
 - Authorization - What can you see/do?
 - Administration - What/where can you manage?

© 2015 IBM Corporation



IBM Cognos BI authentication is based on the use of third party authentication providers. These define users, groups, and roles used for authentication. User names, IDs, passwords, regional settings, and personal preferences are some examples of information stored in the providers.

Authorization is the process of granting or denying access to content, and specifying the actions that can be performed on that content, based on a user identity.

Authorization assigns permissions to users, groups, and roles that allow them to perform actions, such as read or write, on objects, such as folders and reports. Permissions can be granted to users, groups, or roles directly from authentication providers or through membership in Cognos namespace groups and roles.

The Cognos namespace is the built-in namespace from IBM Cognos BI. It contains the IBM Cognos objects, such as groups, roles, data sources, distribution lists, and contacts. During the content store initialization, built-in and predefined security entries are created in this namespace, and include default access to functionality.

You can configure and administer IBM Cognos BI security using IBM Cognos Configuration and IBM Cognos Administration.

Business Analytics software

IBM

IBM Cognos BI Groups and Roles

- IBM Cognos BI provides default groups and roles for security such as:

System Administrators

Authors

Query Users Analysis Users

Consumers

Readers

© 2015 IBM Corporation

Take advantage of IBM Cognos BI groups and roles from the Cognos namespace to secure your IBM Cognos environment and content. The group or role to which a user belongs determines how much access the user has to the IBM Cognos environment and functionality. For example, if you are a member of only the Consumers role, you cannot access any of the IBM Cognos studios.

Besides the default groups and roles, you can create new groups and roles that are specific to your IBM Cognos needs. Simply add users from your authentication source to specific groups and roles as required.

Not only can you use the groups and roles defined in the IBM Cognos namespace to control access to contents, you can use groups in your authentication provider as well.

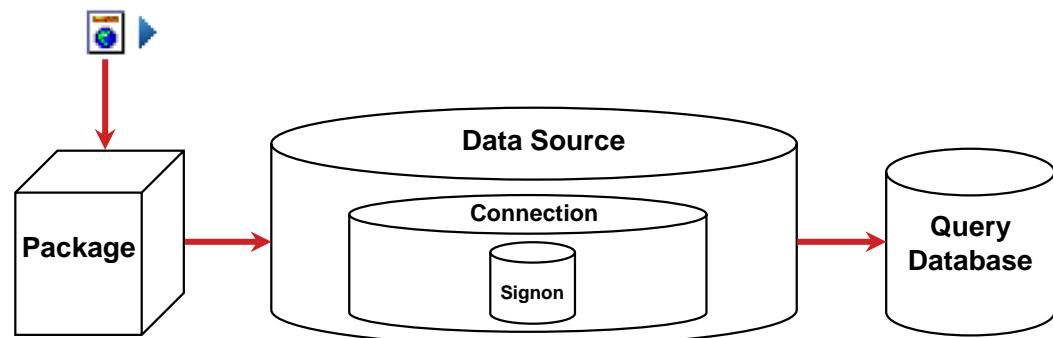
Using the IBM Cognos namespace does not require the IT department and creates a more portable environment.

There are many different groups and roles the administrator can use to restrict what you can see, what you can do, etc.

See the Predefined Entries section of the Administration and Security Guide for detailed information on the predefined groups and roles as well as the anonymous user.

Package/Data Source Relationship

What happens at runtime?



© 2015 IBM Corporation



When a user runs a report, interactively or in the background, the metadata and data in the report are accessed through a combination of the package from which the report was authored, and the data source from which the package was modeled. The data source includes a connection string to the database and may include a signon that allows access to the database. The data source is used to query the database and retrieve the appropriate data, and the result set is presented back to the user.

There may be multiple connections for a given data source and multiple signons for a given connection.

Reports run in the background have saved output. The output is a snapshot of the data at the time the report is run and which is saved in the content store. Note: users have the option to save the report output after running the report interactively. When viewing saved output, there is no querying of the database. Users view the data and metadata from the output version that is saved in the content store.

Each object (report, package, data source, connection, and signon) may have security applied.

Demo 1: Explore IBM Cognos BI

Purpose:

You will obtain a high-level view of how the IBM Cognos BI system works by navigating through the system and tracing the lifecycle of a data item, from its appearance in a report to its existence in an underlying data source. You will take on different IBM Cognos BI roles including a(n):

- Consumer who runs reports or performs analysis to answer business questions
 - Author who creates reports using various data items from a metadata package
 - Data Modeler who imports data from the underlying data source, models it, and publishes a metadata package to make it available for Authors
 - Administrator who creates and manages data source connections
- You will conclude by identifying how the data item from the report appears as a column in the underlying data source.

Task 1. Run a report from IBM Cognos Connection.

1. Open Internet Explorer, and then navigate to <http://localhost:88/ibmcognos>.
2. Log on as scottb/Education1.

The Welcome to IBM Cognos software page appears. Security is currently set in this environment and you have logged on as a member of the Consumers role. As can be seen in the user interface, this role has limited access to functionality within the environment, but at this point you are only interested in running a report to answer your business question. As a Consumer, you have the ability to navigate to and run reports. To do this you will view report content in IBM Cognos Connection.

3. Click **IBM Cognos content**.

You are now in IBM Cognos Connection. This is the portal page for accessing and managing IBM Cognos BI content, including reports, analyses, additional portal pages, and metadata packages for authoring reports. Note: members of the Consumers role can only view packages and cannot use them to author reports. You will see packages as you navigate through the folder hierarchy.

In IBM Cognos Connection, you will find public content, found on the Public Folders tab, or personal content, found on the My Folders tab. You also have the ability to create more personal tabs to suit your needs or to share with others.

Now you want to view and run the report that will answer your business questions.

4. Navigate to **Samples > Models > GO Data Warehouse (query) > Report Studio Report Samples**, and then click the **Total Revenue by Country** report.

Note: GO Data Warehouse is a package as indicated by the package icon which is different from a folder icon.

The report opens in IBM Cognos Viewer.



Before the full report can run, a prompt page appears asking you to provide some contextual information. In this case, you are prompted to select one or more countries for which you want to see revenue data. At runtime, after the Total Revenue by Country link is clicked, the report begins to run and the underlying data source is queried for data. In this case, the query returns data that populates the prompt.

5. Click **Select all**, and then click **Finish**.

The full report runs and opens in IBM Cognos Viewer.

A section of the results appear as follows:

		 Total Revenue by Country For Product Line				
		Revenue	Camping Equipment	Golf Equipment	Mountaineering Equipment	Outdoor Protection
Americas	Brazil					
Americas	Brazil	Ao ar livre	2,554,044.39		2,171,110.26	56,302.39
		Ar fresco	2,551,975.28		3,882,366.19	180,270.92
		Armazém do esporte	1,145,731.4	1,485,312.43		108,891.35
		Casa do Alpinista	1,961,779.8	1,617,961.38		6,813.54
		Esportes Grumari	12,908,332.31	11,256,888.81		665,820.03
		Esportópolis	7,365,113.06	719,281.09		371,442.79
		Galáxia do esporte	2,476,455.78	495,727.77		59,362.27
		Lojas do Esportista	2,788,570.61	1,079,903.72		30,643.97
		Mega Shop do Esporte	273,381.27			8,127

The report contains a second query which returns data that has been filtered, based on information supplied in the prompt.

The report is a crosstab report with rows and columns on the edges and intersecting values in the cells. It also contains:

- a title
- an image that is the company logo
- various report items, such as Revenue, Product lines, Region, Country, Retailer name
- variables that display the report execution time and page numbering (at the bottom of the page)
- multiple pages

6. Scroll through the report.

As a consumer, you can answer business questions, including:

- What is the total revenue per product line in each country?
- What is the total revenue per product line for each retailer?

Take note of the Revenue data item, the values of which populate intersecting cells of the report. This is the item you will trace back to its source. You will assume the role of an author and examine how the report was designed, and how the Revenue data item was used.

7. Click **Bottom**, and then scroll to the end of the report.

This report also includes a chart to provide a visual representation of the data.

8. On the toolbar, click **Return** , and then click **Log Off**.

Task 2. Examine a report in Report Studio.

1. Click **Log on again**.

2. Log on as **brettonf/Education1**.

In the current security environment, Frank Bretton is a member of the Author's role, which by default, provides access to the various reporting and analysis studios within IBM Cognos BI.

3. Click **IBM Cognos content** and then navigate to **Samples > Models > GO Data Warehouse (query) > Report Studio Report Samples**.

4. Beside the **Total Revenue by Country** report, under **Actions**, click **Open with Report Studio - Total Revenue by Country** .

The report opens in design mode in Report Studio, as indicated by the metadata values instead of actual values appearing in the report. Take note of the formatting of the various textual items, including font size and weight. All of these properties can be edited in Report Studio.

5. Click the **Work area**, to the left of the chart, and then from the **Explorer** bar, click **Page Structure** .

The report includes a hierarchical object structure, beginning with a Page object, which includes Page Header, Page Body, and Page Footer objects. The Page Body hierarchy includes, Block objects, which in turn includes Table objects, which in turn includes Table Row objects, etc.

6. From the **Explorer** bar, click **Page Design** .

7. At the bottom of the report, double-click `<%AsOfDate()%>`.

The Report Expression dialog box opens showing the AsOfDate() expression. AsOfDate() is an embedded report function within the Report Studio expression editor, which can be used to return and display the execution date for the report.

8. Click **Cancel**, and then repeat step 7 for `<%PageNumber()%>` and `<%AsOfTime()%>`.

`PageNumber()` returns the current page number.

`AsOfTime()` returns the report execution time.

9. In the crosstab report object, double-click the **Revenue** item.

The objects used to define this expression come from the GO Data Warehouse (query) package, as shown in the Available Components pane on the left. You can see objects from the package by navigating the hierarchy.

10. In the **Available Components** pane, expand **Sales and Marketing (query) > Sales (query) > Sales fact** to locate the **Revenue** object.

The hierarchy you have navigated matches what is displayed in the Expression Definition pane. Note that the Sales and Marketing (query) object is a folder and is excluded from the expression. You will become familiar with this object hierarchy when you create a report in Task 3.

11. Click **Cancel**, and then repeat steps **9** and **10** to identify the expressions and objects used to define them for the following items in the report:

<#Region#>
 <#Retailer country#>
 <#Retailer name (multiscript)#>
 <#Product line#>
 <#Total (Product line)#>
 <#Total (Retailer country)#>

Next you will create a report using metadata objects from a package.

Task 3. Create a report in Report Studio.

1. From the **File** menu, click **New**, click **List**, and then click **OK**.

You are using the List template for creating this report. The template includes the List report object (column headers and columns), Text Item object for the title, and report expressions (at the bottom) for date and time the report is run and page number.

The package that is currently open in Report Studio will, by default remain open for the creation of the report.

Note: To open Report Studio with a different package, you do not have to navigate back to IBM Cognos Connection. Instead, right-click in the Source pane, click Report Package, and then navigate to and open the appropriate package.

The Source pane on the left displays the contents of the GO Data Warehouse (query) package. This package has been published from IBM Cognos Framework Manager as a metadata source for report authors to create reports.

Note: This package is also available to other studios to create reports, including IBM Cognos Workspace Advanced, Query Studio, Analysis Studio, and Event Studio.

The structure and organization has been defined in the IBM Cognos Framework Manager model. There are four folders in this package. Note that this is the same structure that was seen in Task 2, step 9 when you were examining the Revenue item in the Report Expression dialog box.

2. Expand the **Sales and Marketing (query)** folder.

At this level you are viewing namespaces. A namespace provides containment and name qualification for child objects.

3. Expand the **Sales (query)** namespace.

At this level you are viewing query subjects. A query subject is a set of query items that have an inherent relationship. In most cases, query subjects behave like tables. Query subjects produce the same set of rows regardless of which columns were queried.

4. Expand the **Sales fact** query subject.

At this level you are viewing query items. A query item represents a single characteristic of something, such as the date that a product was introduced.

Query items are contained in query subjects or dimensions (if using a dimensional data source). For example, a query subject that references an entire table contains query items that represent each column in the table.

5. Drag **Product line** (from **Products**), **Product type** (from **Products**), and **Revenue** (from **Sales Fact**) query items to the work area.

The results appear as follows:

Product line	Product type	Revenue
<Product line>	<Product type>	<Revenue>
<Product line>	<Product type>	<Revenue>
<Product line>	<Product type>	<Revenue>



6. From the toolbar, click **Run Report**.

The report opens in IBM Cognos Viewer and displays revenue values for all product types from each product line. Next you will examine the SQL that is generated and sent to the data source when the report is run.

7. Close **IBM Cognos Viewer**, and then from the **Tools** menu, click **Show Generated SQL/MDX**.

In Report Studio, you have the option to view either the Native SQL that is sent to and interpreted by the data source or the IBM Cognos SQL that is generated by the IBM Cognos query engine. For the purposes of tracing data access back through the IBM Cognos BI system, you will view the IBM Cognos SQL.

8. In the **Generated SQL/MDX** list, click **IBM Cognos SQL**.

The results appear as follows:

```

Generated SQL

Query results:
  ▾ Query1
    Query1.0

Generated SQL/MDX:
IBM Cognos SQL
select
  Product.Product_line as Product_line,
  Product.Product_type as Product_type,
  XSUM(SLS_SALES_FACT.SALE_TOTAL) for
Product.Product_line,Product.Product_type ) as Revenue
from
  (select
    SLS_PRODUCT_LINE_LOOKUP.PRODUCT_LINE_EN as
Product_line,
    SLS_PRODUCT_TYPE_LOOKUP.PRODUCT_TYPE_EN as
Product_type,
    SLS_PRODUCT_DIM.PRODUCT_KEY as Product_key
  from
    great_outdoors_warehouse..GOSALES DW.SLS_PRODUCT_DIM
    SLS_PRODUCT_DIM,
    great_outdoors_warehouse..GOSALES DW.SLS_PRODUCT_LINE_LOOK
    UP SLS_PRODUCT_TYPE_LOOKUP
  )

```

For the purposes of tracing data access back through the IBM Cognos BI system, you will not examine the SQL in detail, other than to note the following items:

SLS.SALES_FACT.SALE_TOTAL, as Revenue - Revenue is actually a query item called SALE_TOTAL which comes from a query subject named SLS_SALES_FACT. Revenue was renamed from SALE_TOTAL as part of the modeling process in IBM Cognos Framework Manager.

great_outdoors_warehouse.GOSALES DW - at runtime data access is achieved through a data source connection called great_outdoors_warehouse and a schema called GOSALES DW.

9. Click **Close**, and then close **Report Studio** without saving the report.

Next you will take on the role of a modeler/developer to identify how the objects from the package, including the Revenue query item, are made available to authors to create their reports.

Task 4. Examine a model in IBM Cognos Framework Manager.

1. From the **Start** menu, click **All Programs > IBM Cognos 10 > IBM Cognos Framework Manager**.
2. Click **Open a project**, and then open **great_outdoors_warehouse.cpf** from **C:\Program Files(x86)\IBM\cognos\c10\webcontent\samples\models\great_outdoors_warehouse**.
3. Log on as **admin/Education1**.

In the current security environment, Admin Person is a member of the System Administrators role, and by default, has access to the entire IBM Cognos BI system, including IBM Cognos Framework Manager.

You will examine what was published from IBM Cognos Framework Manager.

4. In the **Project Viewer** pane, expand **Packages**.

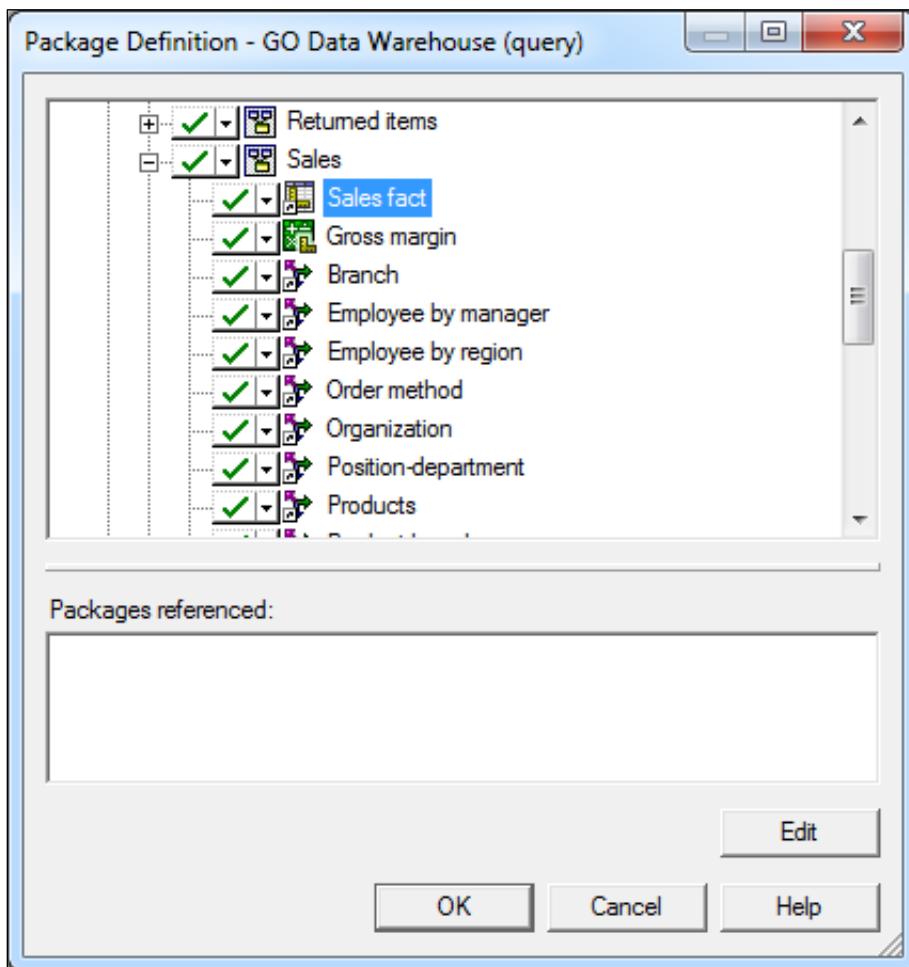
The GO Data Warehouse (query) package appears. This is the package that you used in Report Studio to author the report.

5. Double-click the **GO Data Warehouse (query)** package.

The Package Definition window displays which objects have been included and excluded from the package. The included objects are set to either visible or hidden. Hidden meaning they are included for publishing but will be hidden for report authors.

Note the four folders that have been included and set to visible. These match the folders you identified when you were viewing the package in Report Studio (Task 3, step 1).

6. Expand the **Sales and Marketing (query)** folder > **Sales (query)** namespace.
The results appear as follows:



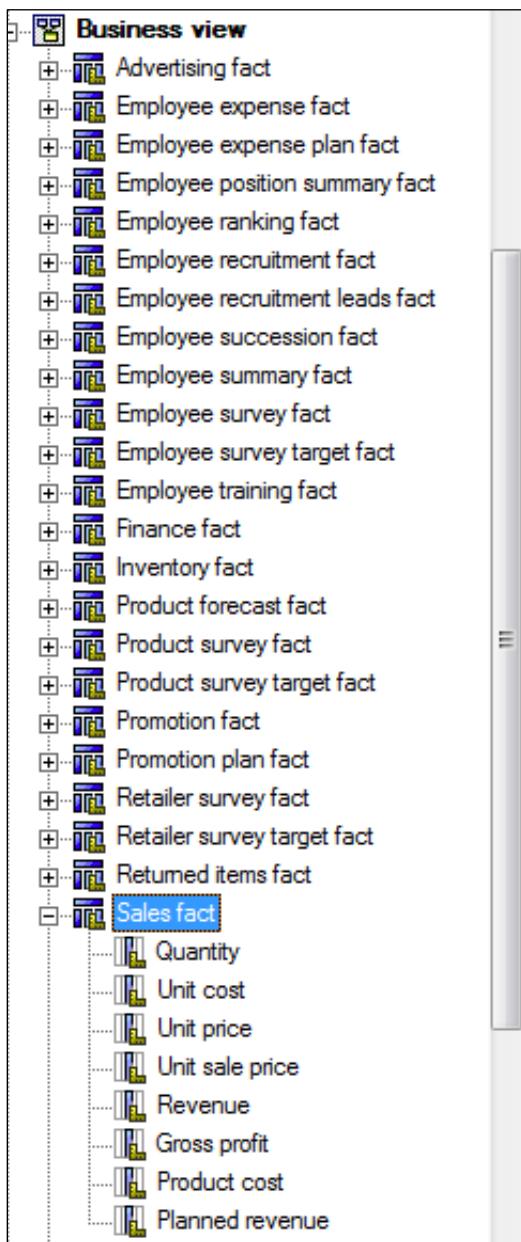
The Sales (query) namespace contains shortcuts to other objects in the model. The Sales fact shortcut points to a source object that is also included in the package; however, you cannot trace the source using this window. To do this, you will examine the model.

7. Click **Cancel**, and then in the **Project Viewer**, expand the **go_data_warehouse** namespace > **Sales and Marketing (query)** folder > **Sales (query)** namespace.

Again, you can see the Sales fact shortcut, but now you can trace that shortcut back to its source.

8. Right-click **Sales fact**, and then click **Go To Target**.

A section of the results appear as follows:



You are taken inside the Business View namespace (bolded text), to the Sales fact query subject (blue background). This query subject includes the Revenue query item.

Without going in to detail on the structure of this model, you should note that, as a proven practice, this sample model has been organized to include multiple namespaces. The Database View namespace is used to contain query subjects and query items that have been imported directly from the data source. The Business View namespace is used to contain query subjects and query items on which various modeling tasks have been performed, such as:

- combining query items from multiple query subjects
- creating calculated query items
- creating model filters and query item filters
- modifying and creating relationships between query subjects
- setting of various query subject and query item, properties

Objects that will be made available to report authors are typically kept outside of these namespaces, for example the Sales and Marketing (query) folder.

The Sales and Marketing (query folder) contains only shortcuts to objects elsewhere in the model. Some of those objects, contained elsewhere in the model, have been included in the package. In the Package Definition, you noted that some objects were made hidden, including the Business View namespace. In Report Studio, when the author creates the report and uses the Revenue item, they are using a shortcut to a hidden object in the package. In this case, it is the Revenue query item, from the Sales fact query subject, in the Business View namespace.

You will now identify the source for the Revenue query item from the Sales fact query subject.

9. Double-click the **Sales fact** query subject.

This is a model query subject. Model query subjects can be used to create a more abstract, business-oriented view of a data source. For example, you can add business objects such as calculations and filters and combine query items from other query subjects, including other model query subjects.

The Query Items and calculations pane displays the query items and their source that make up this model query subject, including the Revenue query item. Also note that there are two calculated items: Product cost, and Planned revenue. For Revenue, there are two items to note:

- The source is SLS_SALES_FACT.SALE_TOTAL.
- The name is Revenue, indicating that the item was renamed from SALE_TOTAL

From this, you can conclude that Revenue is sourced from SALE_TOTAL. To locate this object, you can search for it in the model.

10. Click **Cancel**, and then in the **Tools** pane on the right, click the **Search** tab.
11. Search for **SALE_TOTAL** using the **go_data_warehouse** model as the scope.

Tip: Use the Search options  button to define your search.

12. Click the first instance that contains **SALE_TOTAL**.

Objects in the Project Viewer pane expand, and you can see the SALE_TOTAL query item is located at go_data_warehouse > Database view > Sales and marketing data > SLS_SALES_FACT.

13. Double-click **SLS_SALES_FACT**.

This is a data source query subject. Data source query subjects directly reference data in a single data source. IBM Cognos Framework Manager automatically creates a relational data source query subject for each table and view that you import into your model. It includes an SQL statement that will, at runtime, retrieve all the columns from the table. You will test this behavior and in turn locate the SALE_TOTAL column and its values.

14. Click the **Test** tab, click **Test Sample**, and then in the **Test results** window, scroll to the right to locate **SALE_TOTAL**.

SALE_TOTAL is returned as a column.

15. Click the **Query Information** tab.

The results appear as follows:

The screenshot shows the 'Query Subject Definition - SLS_SALES_FACT' dialog box. The 'Query Information' tab is selected. The 'Cognos SQL' section contains the following select statement:

```

select
    SLS_SALES_FACT.SALES_ORDER_KEY as SALES_ORDER_KEY,
    SLS_SALES_FACT.ORDER_DAY_KEY as ORDER_DAY_KEY,
    SLS_SALES_FACT.EMPLOYEE_KEY as EMPLOYEE_KEY,
    SLS_SALES_FACT.ORGANIZATION_KEY as ORGANIZATION_KEY,
    SLS_SALES_FACT.RETAILER_SITE_KEY as RETAILER_SITE_KEY,
    SLS_SALES_FACT.PRODUCT_KEY as PRODUCT_KEY,
    SLS_SALES_FACT.PROMOTION_KEY as PROMOTION_KEY,
    SLS_SALES_FACT.ORDER_METHOD_KEY as ORDER_METHOD_KEY,
    SLS_SALES_FACT.RETAILER_KEY as RETAILER_KEY,
    SLS_SALES_FACT.SHIP_DAY_KEY as SHIP_DAY_KEY,
    SLS_SALES_FACT.CLOSE_DAY_KEY as CLOSE_DAY_KEY,
    SLS_SALES_FACT.QUANTITY as QUANTITY,
    SLS_SALES_FACT.UNIT_COST as UNIT_COST,
    SLS_SALES_FACT.UNIT_PRICE as UNIT_PRICE,
    SLS_SALES_FACT.UNIT_SALE_PRICE as UNIT_SALE_PRICE,
    SLS_SALES_FACT.GROSS_MARGIN as GROSS_MARGIN,
    SLS_SALES_FACT.SALE_TOTAL as SALE_TOTAL,
    SLS_SALES_FACT.GROSS_PROFIT as GROSS_PROFIT
from
    great_outdoors_warehouse..GOSALEDW.SLS_SALES_FACT SLS_SALES_FACT

```

The 'from' clause is highlighted with a red box. The 'Native SQL' section contains the following select statement:

```

select "SLS_SALES_FACT"."SALES_ORDER_KEY" "SALES_ORDER_KEY", "SLS_SALES_FACT"."ORDER_DAY_KEY"
"ORDER_DAY_KEY", "SLS_SALES_FACT"."EMPLOYEE_KEY" "EMPLOYEE_KEY", "SLS_SALES_FACT"."ORGANIZATION_KEY"
"ORGANIZATION_KEY", "SLS_SALES_FACT"."RETAILER_SITE_KEY" "RETAILER_SITE_KEY",
"SLS_SALES_FACT"."PRODUCT_KEY" "PRODUCT_KEY", "SLS_SALES_FACT"."PROMOTION_KEY" "PROMOTION_KEY",
"SLS_SALES_FACT"."ORDER_METHOD_KEY" "ORDER_METHOD_KEY", "SLS_SALES_FACT"."RETAILER_KEY" "RETAILER_KEY",
"SLS_SALES_FACT"."SHIP_DAY_KEY" "SHIP_DAY_KEY", "SLS_SALES_FACT"."CLOSE_DAY_KEY" "CLOSE_DAY_KEY",
"SLS_SALES_FACT"."QUANTITY" "QUANTITY", "SLS_SALES_FACT"."UNIT_COST" "UNIT_COST",
"SLS_SALES_FACT"."UNIT_PRICE" "UNIT_PRICE", "SLS_SALES_FACT"."UNIT_SALE_PRICE" "UNIT_SALE_PRICE",
"SLS_SALES_FACT"."GROSS_MARGIN" "GROSS_MARGIN", "SLS_SALES_FACT"."SALE_TOTAL" "SALE_TOTAL",
"SLS_SALES_FACT"."GROSS_PROFIT" "GROSS_PROFIT" from "GOSALEDW"."SLS_SALES_FACT" "SLS_SALES_FACT" FOR FETCH
ONLY

```

As with the Generated SQL/MDX window in Report Studio, this window also displays the SQL that is sent to the data source (Native SQL) and the SQL generated by the IBM Cognos query engine (Cognos SQL). In the Cognos SQL, you can see that the select statement includes all columns from the SLS_SALES_FACT table, including the SALE_TOTAL column. You can see that the "from" statement includes an object called great_outdoors_warehouse, and one called GOSALEDW. These objects represent the data source connections that are used at runtime. You will examine how these connections are defined in Task 5 when you view them in IBM Cognos Administration. For now, you will continue to examine how they are used in IBM Cognos Framework Manager.

This material is meant for IBM Academic Initiative use only. NOT FOR RESALE

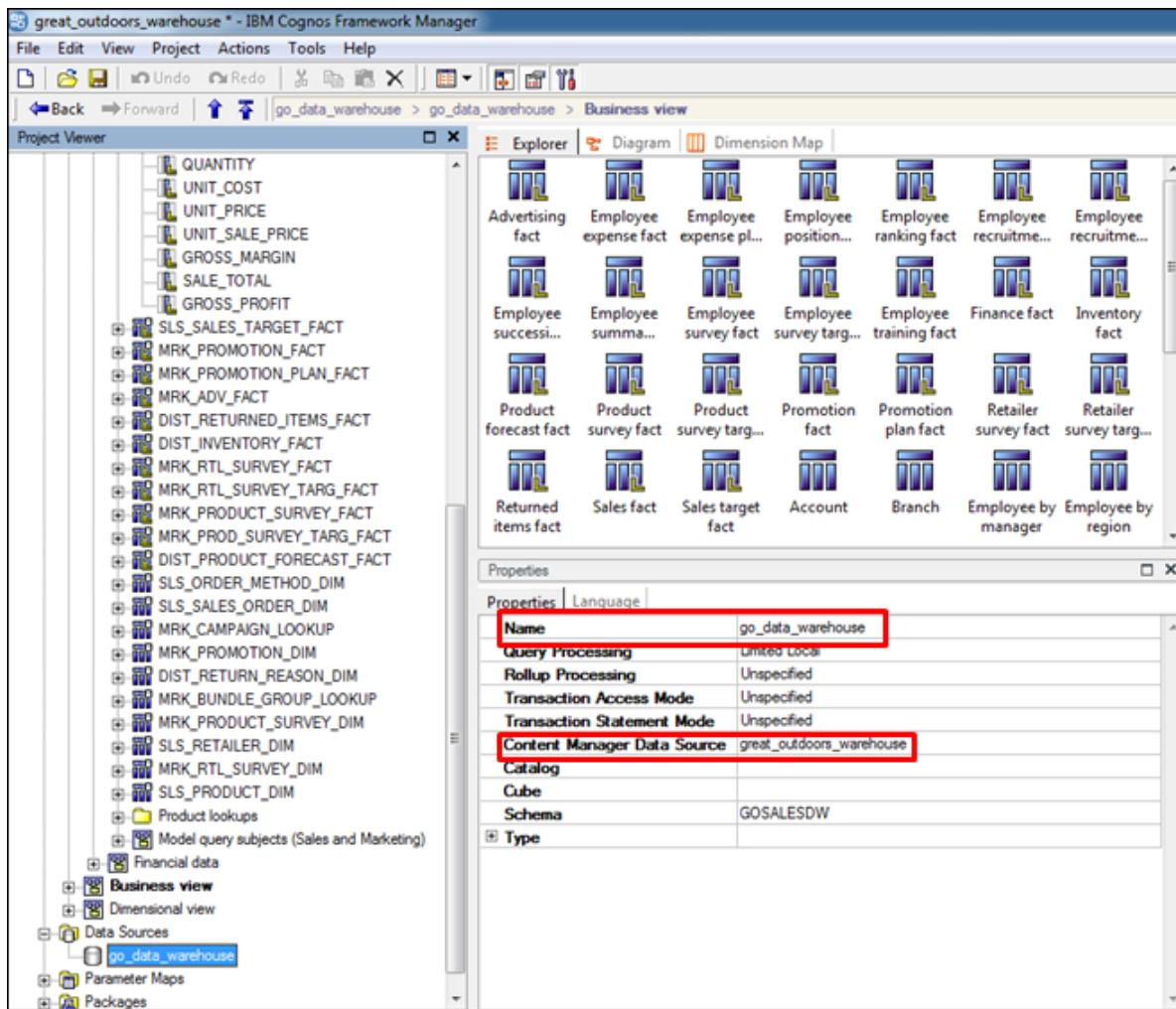
© 2003, 2015, IBM Corporation

This guide contains proprietary information which is protected by copyright. No part of this document may be photocopied, reproduced, or translated into another language without a legal license agreement from IBM Corporation.

1-29

16. Click **Cancel**, and then in the **Project Viewer**, expand **Data Sources**, and then click **go_data_warehouse**.

The results appear as follows:



From the information in the Properties pane, you can see that this model makes use of data source named **go_data_warehouse**, which uses a Content Manager data source named **great_outdoors_warehouse**. In this case, the modeler has opted to provide a different name for the model data source by editing the **Name** property in the Properties pane.

The Content Manager Data Source is the object through which:

- runtime data access is achieved
- the process of importing data source objects into the model is accomplished

You will examine the import process by running the Metadata Wizard.

17. In the **Project Viewer**, right-click **Database view**, and then click **Run Metadata Wizard**.

You can import from a wide range of data source types.

18. Leave **Data Sources** selected, and then click **Next**.

There are currently two or three data sources to choose from to perform the import. Clicking the New button will let you create a new data source provided you have the appropriate access rights to perform this action.

One of the data sources is named great_outdoors_warehouse and is the data source that was previously used to import objects in to this model. The name also matches the Content Manager data source name identified at step 16.

19. Click **great_outdoors_warehouse**, and then click **Next**.

This data source points to a set of child data source objects, one of them being GOSALESDW. This name matches the object name identified when examining the SQL statement in step 15.

20. Expand **GOSALESDW > Tables**.

This data source object includes a set of tables, which can be imported into the model. Included in this set is the SLS_SALES_FACT table.

21. Expand **SLS_SALES_FACT**.

This table includes a set of columns, which can be imported into the model. Included in this set is the SALE_TOTAL column.

22. Click **Cancel**, and then close **IBM Cognos Framework Manager** without saving the project.

Up to this point you have identified how the Revenue query item:

- data values appear in an existing report in IBM Cognos Viewer
- is used in a new report created in Report Studio
- is made available to report authors in a package that is published from IBM Cognos Framework Manager
- is sourced and modeled in the IBM Cognos Framework Manager model
- is imported into the IBM Cognos Framework model as the SALE_TOTAL column

23. In **IBM Cognos connection**, click **Log Off**.

Task 5. Examine data sources in IBM Cognos Administration.

You will now take on the role of the administrator to examine how data source connections are defined in IBM Cognos Administration.

1. Click **Log on again**.
2. Log on as **admin/Education1**, and then on the Welcome to IBM Cognos Software page, click **Administer IBM Cognos content**.

IBM Cognos Administration is the portal that allows you to monitor and administer the IBM Cognos BI system including servers, security, capabilities, data source connections, and the deployment of content. In the current security environment, Admin is a member of the Directory Administrators role, which by default, provides access to the Directory pages of IBM Cognos Administration, including the ability to create and manage data sources.

3. Click the **Configuration** tab.

The first node selected is Data Source Connections. Here you can administer existing data sources and create new ones. Note: The New Data Source button on the toolbar provides the same user interface experience for creating a new data source that is available in IBM Cognos Framework Manager in the Metadata Wizard.

Existing data sources display; these are the same data sources that appeared when you ran the Metadata Wizard in IBM Cognos Framework Manager, at Task 4, steps 17 and 18.

4. Click the **great_outdoors_warehouse** data source.

This data source includes a single data source connection named **great_outdoors_warehouse**. Note: You can have multiple data source connections for a single data source. For example, if you have multiple databases with exactly the same structure (but different data), you can create one data source with multiple connections. The data source connection identifies which database you want to connect to.

5. Under **Actions**, click **Set properties - great_outdoors_warehouse**, and then click the **Connection** tab.

This connection is configured to connect to a DB2 database. There are many types available for creating connections.



6. Beside the **Connection string** box, click **Edit the connection string**.
The connection is to a database named GS_DB, and under Signon, a signon has been configured for this connection. You will now examine the signon.
7. Click **Cancel** twice, and then click the **great_outdoors_warehouse** connection.
This connection includes a single signon named great_outdoors_warehouse. The database signon identifies the user's rights in the database. You can have multiple database signons that have access to different tables. Within the database, you can create sets of tables with different owners or schemas, and then provide access to these with the appropriate signon.
8. Under **Actions**, click **Set properties - great_outdoors_warehouse**, click the **Signon** tab, and then click **Edit the signon**.
This signon is configured to connect to the GOSALES DW schema using the credentials of the GOSALES DW user. The GOSALES DW schema is the object referenced in the generated SQL when you examined it in Report Studio (Task 3, step 8) and in IBM Cognos Framework Manager (Task 4, step 15)
9. Click **Log Off**, and then close **Internet Explorer**.
You have identified how the connection is made to the underlying data source. Next, you will examine the required data source objects as they appear in IBM DB2.

Task 6. Examine underlying data source objects.

1. From the **Start** menu, navigate to **All Programs>IBM Data Studio>Data Studio 4.1.0.0 Client**.
2. Click **OK** to close the **Workspace Launcher**.
3. From the **Administrator Explorer** pane, expand **localhost>50000** and then double-click **GS_DB [DB2 Alias]**.
4. In the **Properties for GS_DB** window, on the **General** tab, in the **User name** field type **db2admin**, and then in the **Password** field type **Education1**.
5. Click the **Save password** check box, and then click **OK**.
6. Expand **GS_DB**.

The GS_DB database connection is active and folders are displayed. This is the database identified at Task 5, step 6.

7. From the **Administrator Explorer** pane, double-click the **Schemas** folder.
- The Schemas folder is displayed in the central pane. This is the schema that the great_outdoors_warehouse data source is connecting to as identified in Task 5, step 8.

A section of the results appear as follows:

Name	Owner
DB2ADMIN	SYSIBM
GOSALES	SYSIBM
GOSALESDW	SYSIBM
GOSALESHR	SYSIBM
GOSALESMR	SYSIBM
GOSALESRT	SYSIBM
NULLID	SYSIBM
SQLJ	SYSIBM
SYSCAT	SYSIBM
SYSFUN	SYSIBM
SYSIBM	SYSIBM
SYSIBMADM	SYSIBM
SYSIBMINTERNAL	SYSIBM

8. From the **Administrator Explorer** pane, double-click the **Tables** folder.
 9. In the **Task Launcher** pane scroll down to **SLS_Sales_Fact** (name column).
 10. Right-click **SLS_Sales_Fact**, point to **Data**, and then click **New "Select" Script**.
 11. From the toolbar click **Run SQL** .
- The query executes and returns data from all the columns in the table.
12. From the **Properties** pane, at the bottom of the pane, click the **Result1** tab on the lower right pane.

13. Scroll to the right to locate the **SALE_TOTAL** column and its values.
You have traced the Revenue item in the Total Revenue by Country report all the way back to its source in the underlying data source and have identified how the IBM Cognos BI system works.
14. Close all open windows.

Results:

You obtained a high-level view of how the IBM Cognos BI system works by navigating through the system and tracing the lifecycle of a data item, from its appearance in a report back to existence in an underlying data source.

Extend IBM Cognos BI Enterprise

- IBM Cognos provides a wide variety of ways to extend IBM Cognos BI.
- For more information, please visit the [IBM Cognos Web site](#)
[http://www-01.ibm.com/software/analytics/cognos/.](http://www-01.ibm.com/software/analytics/cognos/)

© 2015 IBM Corporation



Cognos products that extend IBM Cognos BI include:

- IBM Cognos for Microsoft Office (integrate IBM Cognos content with MS Office)
- IBM Cognos Mobile (IBM Cognos content on mobile devices)
- IBM Cognos Analysis for Microsoft Excel (multidimensional analysis on IBM Cognos BI data in MS Excel spreadsheets)
- IBM Cognos Mashup Service

For developers, there is also Composite, which allows for access to an even wider variety of data sources, and the IBM Cognos SDK for customization and application development.

For those requiring access to real-time monitoring of operational data, IBM Cognos offers IBM Cognos BI Real-time Monitoring, which delivers highly visual, interactive, and self-service dashboards, data integration, analysis, and reports.

You can incorporate TM1 widgets into IBM Cognos to allow users to interact with financial plans.

This material is meant for IBM Academic Initiative use only. NOT FOR RESALE

Summary

- At the end of this module, you should be able to:
 - describe IBM Cognos Business Intelligence (BI) and its position within the IBM Smarter Analytics approach and offerings
 - describe the IBM Cognos 10 Family of offerings
 - describe IBM Cognos BI enterprise components
 - describe IBM Cognos architecture at a high level
 - describe IBM Cognos BI security at a high level
 - explain how to extend IBM Cognos BI

© 2015 IBM Corporation



This material is meant for IBM Academic Initiative use only. NOT FOR RESALE

© 2003, 2015, IBM Corporation

This guide contains proprietary information which is protected by copyright. No part of this document may be photocopied, reproduced, or translated into another language without a legal license agreement from IBM Corporation.



Identify Common Data Structures

IBM Cognos BI

Business Analytics software



© 2015 IBM Corporation

This material is meant for IBM Academic Initiative use only. NOT FOR RESALE

© 2003, 2015, IBM Corporation

This guide contains proprietary information which is protected by copyright. No part of this document may be photocopied, reproduced, or translated into another language without a legal license agreement from IBM Corporation.

Objectives

- At the end of this module, you should be able to:
 - define the role of a metadata model in Cognos BI
 - distinguish the characteristics of common data structures
 - understand the relative merits of each model type
 - examine relationships and cardinality
 - identify different data traps
 - identify data access strategies

© 2015 IBM Corporation

This module will focus on industry standard data structures. This knowledge is essential before beginning a metadata modeling project. When referring to a modeler in this module, we are referring to a data modeler, not a metadata modeler.

Business Analytics software

IBM

Examine the Role of an IBM Cognos Metadata Model

- An IBM Cognos metadata model provides a business presentation view of an organization's data sources.
- BI users use the model to analyze and report on their data sources.

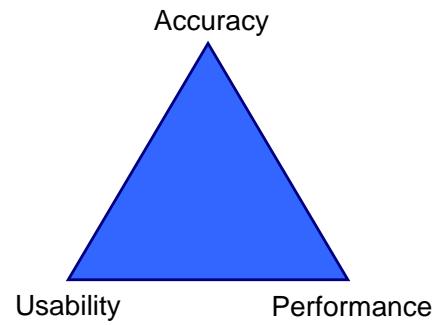
© 2015 IBM Corporation

A metadata model can hide the structural complexity of your underlying data sources. By creating a metadata model, you have more control over how your data is presented to end users. You can also choose which data to display to your end users and how it will be organized. The overall goal of modeling the metadata is to create a model that provides predictable results as well as an easy-to-use view of the metadata for authors and analysts.

Your underlying data sources may be very diverse. For example, you may have operational or reporting data in one or more relational databases. You may also have legacy data in various file formats, such as text, comma separated values (.csv), and extensible markup language (XML). You may even have online analytical processing (OLAP) sources that include cubes (such as Cognos PowerCubes), as well as other sources such as SAP BW.

Goals of a Data Modeler

- Data modelers have 3 main goals:
 - Accuracy: Reports must contain accurate data.
This is the most essential goal.
 - Usability: Packages must be understandable by report authors and other users.
 - Performance: Reports must take the least system resources possible.



Usability and performance offer a trade-off.
Modelers can gain usability by sacrificing performance and vice-versa.



© 2015 IBM Corporation

The goals of the data modeling process can be summarized in three categories:

- **Accuracy:** If the packages created from a model do not produce reports with accurate data, the model is inherently flawed and must be repaired.
- **Usability:** The packages produced from a model must be understandable by those who will use the packages to create reports and perform analysis.
- **Performance:** Data should be retrievable in a reasonable amount of time.

Usability concerns can vary depending on the database understanding and skill level of your authors and analysts. Usability will not always affect performance, but it can. Several modeling techniques can slow down data retrieval. As a modeler, you must determine whether usability or performance is a higher priority for your authors and analysts. Remember, accuracy is not negotiable.

We will examine various methods you can use to improve the performance of your IBM Cognos data retrieval later in this course.

Data Sources and Model Types

- You can work with both relational and dimensional data sources in IBM Cognos BI.
- There are several different model types that support these data sources:
 - relational (operational and reporting models)
 - dimensionally modeled relational (DMR)
 - online analytical processing (OLAP)
 - multidimensional OLAP (MOLAP)
 - relational OLAP (ROLAP)

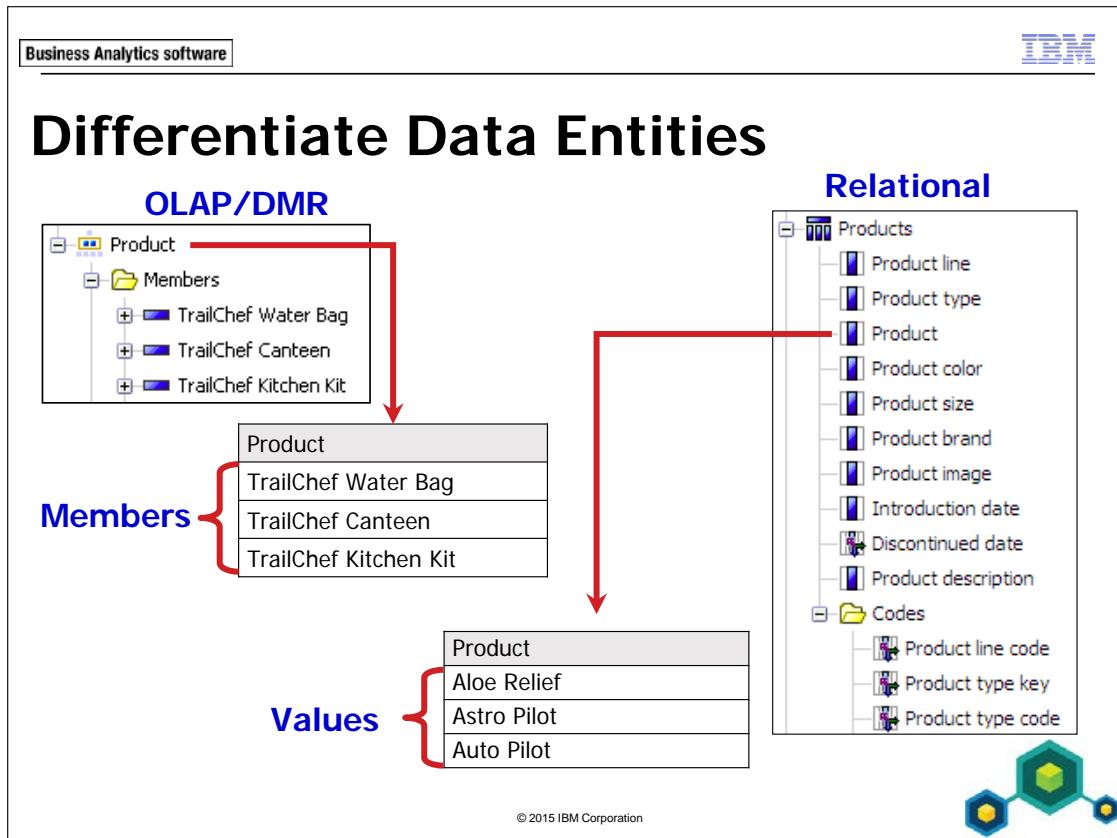
© 2015 IBM Corporation



Relational models have a basic metadata structure that looks like tables and columns in a database. Relational models can have either an operational (normalized) or reporting structure (star-schema). You can use Framework Manager to transform metadata from an operational data source into a model that optimizes it for reporting.

Dimensionally Modeled Relational (DMR) models are built from relational data sources, but are modeled with a dimensional structure (like OLAP) consisting of dimensions, hierarchies, and measures. Using Framework Manager, you can add dimensional metadata to model objects, which enables drill-through capabilities in Cognos BI.

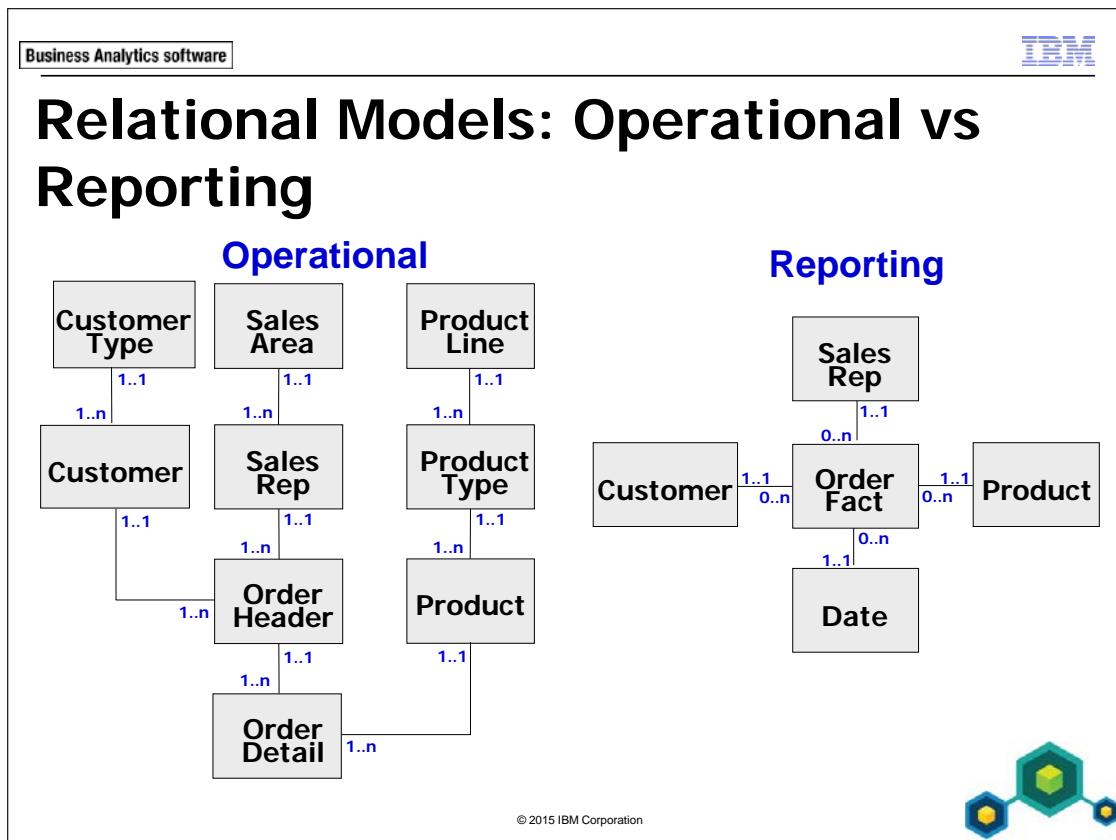
OLAP models point to dimensional data sources, such as IBM Cognos PowerCubes, which are built using their own modeling tools, such as Transformer.



OLAP and DMR data entities are different from relational data entities:

- values (relational)
- members (OLAP and DMR)

When you create a report using a dimensional model, you work with the member. Each member has certain properties such as a member key and member caption. Report authors and consumers see the member caption. Each member is defined and identified by its member unique name (MUN), which describes its position in the dimensional structure.



Operational databases are:

- used to track the day-to-day operations of a business
- usually normalized or part of an enterprise resource planning (ERP) vendor package

Reporting databases are:

- typically a copy of the operational data
- structured to make reporting faster and easier
- usually dimensional, taking the form of a star schema design

In general, we recommend that you create a logical model that conforms to star schema concepts. This is a requirement for IBM Cognos Analysis Studio and has also proved to be an effective way to organize data for your users.

Examining Operational Databases

Operational databases:

- are designed to maximize accuracy and minimize redundancy
- are optimized for writing and updating data rather than reading data
- often result in monolithic designs with multiple joins
- impact the speed of large queries

© 2015 IBM Corporation



It can be difficult to report from an operational database because:

- reading the data can impact the performance of the system
- the number of tables that must be joined to satisfy a business question can be prohibitively large
- a more complex database requires more metadata modeling efforts to make it consumable by business users

Operational systems are designed with one goal in mind: to get data into the database quickly. These databases are normalized to reduce redundancy. Having little to no redundancy ensures that there is data integrity and that database triggers function properly, so that the right data is captured.

Business Analytics software

IBM

Example: Operational Database Query

- "Show all customer types that bought from a product line."
- The query must check data in seven tables before returning a result set.

```

classDiagram
    class CustomerType
    class Customer
    class SalesArea
    class SalesRep
    class ProductLine
    class ProductType
    class OrderHeader
    class OrderDetail

    CustomerType "1..1" --> "1..n" Customer : 1..1
    CustomerType "1..1" --> "1..n" SalesArea : 1..1
    CustomerType "1..1" --> "1..n" ProductLine : 1..1
    Customer "1..n" --> "1..1" CustomerType : 1..1
    Customer "1..n" --> "1..1" SalesArea : 1..n
    Customer "1..n" --> "1..1" ProductLine : 1..n
    SalesArea "1..n" --> "1..1" CustomerType : 1..1
    SalesArea "1..n" --> "1..1" SalesRep : 0..n
    SalesArea "1..n" --> "1..1" ProductLine : 1..1
    SalesRep "0..n" --> "1..1" SalesArea : 1..1
    SalesRep "1..1" --> "1..1" OrderHeader : 1..1
    ProductLine "1..1" --> "1..n" CustomerType : 1..1
    ProductLine "1..1" --> "1..n" ProductType : 1..n
    ProductType "1..n" --> "1..1" ProductLine : 1..1
    ProductType "1..n" --> "1..1" Product : 1..n
    OrderHeader "1..1" --> "1..1" OrderDetail : 1..1
    OrderDetail "0..n" --> "1..1" OrderHeader : 0..n
    OrderDetail "1..1" --> "1..1" Product : 1..1
  
```

Query requires 7 tables

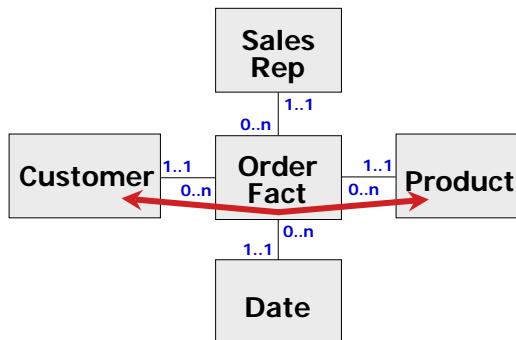
© 2015 IBM Corporation

Operational databases may be too complex for general reporting. As a result, reports that are generated against these structures may take a long time to run. They may also have unpredictable results.

As shown in the slide example, a report may need to access many tables to retrieve all the necessary data.

Example: Reporting Database Query

- Transactional data is stored in a fact table
- Reference data is stored in separate dimension tables



The same query only requires 3 tables



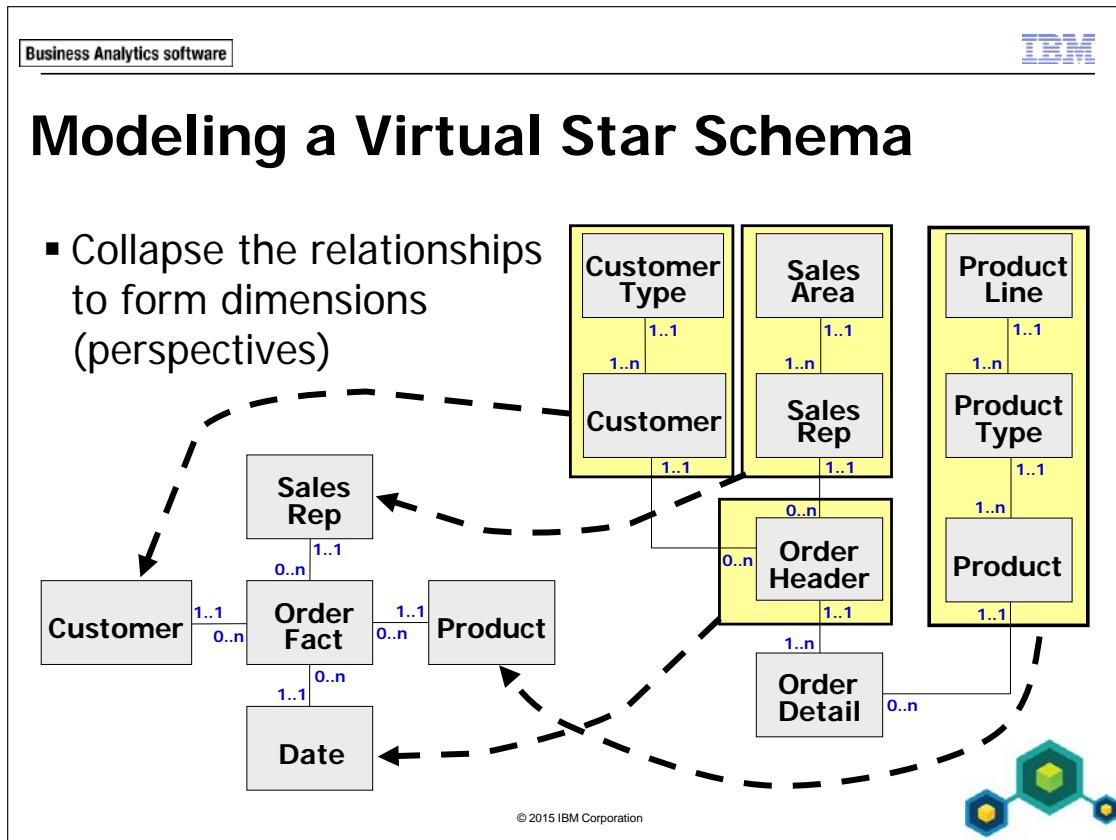
© 2015 IBM Corporation

A star schema is a design with two basic components:

- a central table, known as the fact table (typically numeric data)
- satellite tables, known as the dimension tables

Because a star schema database contains fewer tables than a fully normalized database, query performance is much faster.

- A typical query against a star schema database focuses on the central fact table and makes integrity checks against the related dimension tables.
- If the query is to retrieve information about a specific subject area only, such as all the products that belong to a particular product line, then the query will be even faster.



Each table in a star schema database will contain an expanded set of data.

Extract Transform and Load (ETL) tools can be used to create a star schema data warehouse, or you may use a metadata modeling tool to emulate a star schema structure by generating the appropriate SQL at report design time. The second option will not improve performance, but will yield predictable results.

The Cognos BI ETL tool is called Cognos Data Manager. Framework Manager cannot create a warehouse, but it can emulate a star schema structure by collapsing query subjects to simplify the view and generate the appropriate SQL at run time.

The Data Warehouse Toolkit by Ralph Kimball is a good resource for information on building data warehouses.

Examining Operational Data

- Data is normalized

Product Line Table

PL#	PL_Desc
a	Classic Tents
b	Moose Boots

2 rows

Product Type Table

PL#	PT#	PT_Desc
a	1	Pup Tents
a	2	Family Tents
b	11	Child Boots
b	12	Adult Boots

4 rows

Product Table

PT#	Prod#	Prod_Desc
1	101	1 Sleeper
1	102	2 Sleeper
2	201	4 Sleeper
2	203	6 Sleeper
11	1101	Wet Proof
12	1102	Hikers+

6 rows

Before collapsing into a star schema dimension

© 2015 IBM Corporation



The slide example shows three normalized tables that represent three hierarchical levels. Products roll up into product types, and product types roll up into product lines.

The Product Line table has two rows that indicate two lines of products sold by the company.

The Product Type table contains four rows to indicate the four types of products that fall under the previous two product lines (two types per product line).

The Product table contains the greatest level of detail. It holds 6 rows to represent the 6 products that fall under the four product types.

Business Analytics software

IBM

Examining Reporting Data

- Data is de-normalized

Product Dimension Table

PL#	PL_Desc	PT#	PT_Desc	Prod#	Prod_Desc
A	Classic Tents	1	Pup Tents	101	1 Sleeper
A	Classic Tents	1	Pup Tents	102	2 Sleeper
A	Classic Tents	2	Family Tents	201	4 Sleeper
A	Classic Tents	2	Family Tents	203	6 Sleeper
B	Moose Boots	11	Child Boots	1101	Wet Proof
B	Moose Boots	12	Adult Boots	1102	Hikers

6 rows

After collapsing into a star schema dimension

© 2015 IBM Corporation



The slide example shows a de-normalized dimension table created from the three normalized tables shown on the previous slide.

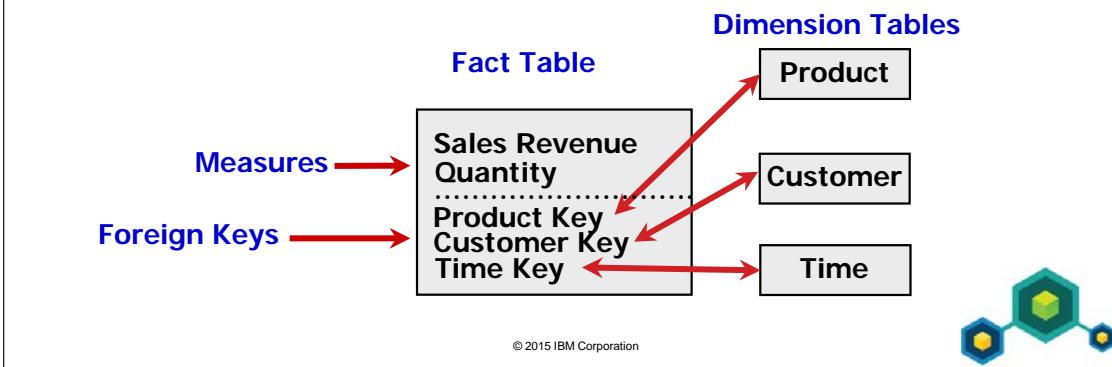
The Product Line table forms the first two columns of the new dimension table (PL# and PL_Desc), the Product Type table forms the next two columns (PT# and PT_Desc), and the Product table forms the last two columns (Prod# and Prod_Desc).

The main characteristic of this table is its redundancy. Note that each product line (Classic Tents and Moose Boots) is repeated, once for each product that the product line contains. The same applies for product type.

This type of table is unsuitable for a normalized system, but is ideal for a reporting and querying structure.

Examining Fact Tables

- Fact tables contain the values (usually additive) by which a company measures itself:
 - Standard Selling Price - not additive
 - Sale Amount - additive



Fact tables are the focal point of any star schema, and typically contain the most rows. There are typically no descriptive attributes in a fact table. Instead, there are foreign keys that relate to the dimension tables, which contain descriptive attributes.

Facts in a fact table are also known as metrics, measures, or key performance indicators. There are cases where you may encounter factless fact tables, in which only foreign keys are found. For example, in the case of a library, you may have a fact table that only contains a book key, a customer key, and a day key, which records which books were checked out by which customer and when.

Business Analytics software

IBM

Examining Dimension Tables

- Dimension tables provide descriptive information.
- Dimension tables may be "conformed" so that they apply to multiple fact tables across the business.

```

graph TD
    Customer[Dimension Customer] --> Sales[Fact Sales]
    Sales --> Product[Dimension Product]
    Sales --> Inventory[Fact Inventory]
    Inventory --> Product
    Customer --> Time[Dimension Time]
    Time --> Product
    Time --> Inventory
    subgraph ConformedDimensions [Conformed Dimensions]
        Customer
        Time
    end

```

© 2015 IBM Corporation

A conformed dimension is one which is common to several fact tables; it has the same meaning and content when being referred to from several fact tables. Conformed dimensions prevent "islands of information" by providing context to multiple potential queries.

In the example above, you can query either sales or inventory data through the Product dimension table, the Time dimension table, or both. For example, Product acts as a context when you want to compare quantity sold with stock count.

Defining Relationships

- Specify how data in one table is linked to data in another table.
- Relationships are implied in the physical data (modeler explicitly declares these relationships)
- Modeler formulates the reality of the business by configuring the relationships

© 2015 IBM Corporation



A relationship states a connection or an operational business rule, such as:

- a sales representative sells a product
- an employee is assigned to a department

Relationships work in both directions. You often have to examine both directions to fully understand the relationship. For example:

- a branch is composed of employees
- an employee may work directly for a branch

Issues with Star Schemas

- Data is only as current as the last data load.
- Structural issues:
 - the distinct count problem
 - very large dimension tables
 - snowflakes
- Fact issues:
 - different levels of granularity (detail) in fact tables

© 2015 IBM Corporation



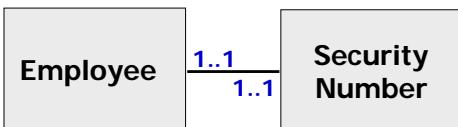
Unlike an operational database, the data in a reporting database (star schema) is not live. Data is periodically loaded from an operational system into the reporting database (star schema layout). Therefore the data is only as current as the last time it was loaded.

When reporting against a star-schema database, you may encounter difficulties in counting the exact number of items in the dimension, such as separate products.

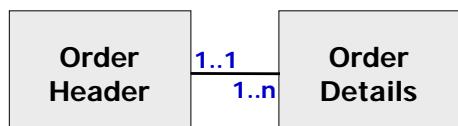
The star schema may also include large dimension tables that result in reports that run too slowly. These tables may be broken out into smaller tables through a normalization process, which in turn creates snowflake tables.

In some cases, fact tables can have different levels of granularity. For example the data values may either be associated with the day level or the month level. The granularity may change based on the dimensions referenced in the fact table.

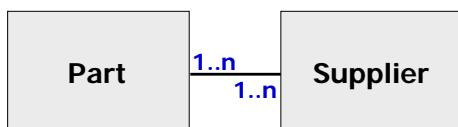
Examining Relationships: Cardinality



- One-to-One: One employee holds exactly one security number.



- One-to-Many: Each order header must have one or more order details.



- Many-to-Many: Each part may be provided by many suppliers, and each supplier may provide many parts.

© 2015 IBM Corporation



Cardinality indicates the number of instances of an entity in relation to another entity.

One-to-one relationships occur when one unique row in a table relates to exactly one row in another table. For example, each employee can only have one security number. One security number can only be associated with one employee.

One-to-many relationship example: an order is taken and an Order Header table is populated with data such as date, customer name, and sales staff name. This table is related to an Order Details table that contains data about individual items sold in that one order, such as order detail code, product number, and quantity. Therefore a relationship exists between Order Header and Order Details, whereby each Order Header must contain one or many Order Details, and each Order Detail must appear on one and only one Order Header.

Many-to-many relationships occur when many unique rows in a table relate to many rows in another table. The slide example shows that many suppliers may provide a single part. However, a single supplier may provide many parts.

Business Analytics software

IBM

Optional vs Mandatory Cardinality

- Mandatory relationship: a row of data (i.e. a product) must exist, in order for a row of data in another table (i.e. an order) to exist.
- Optional relationship: a row of data (i.e. a sale) does not have to exist in order for a row of data in another table (i.e. sales rep) to exist.

© 2015 IBM Corporation

You can specify an optional relationship (minimum cardinality of 0) when you want the query to retain the information on the other side of the relationship in the absence of a match. An optional relationship generates an outer join, which results in a null value when there is no data for one table that matches a row of data in another table.

Make sure that you only define optional relationships when required, as generating outer joins can negatively affect performance.

Examining Data Traps

- There are four basic data traps:
 - chasm trap (many-to-many relationship)
 - transitive relationship trap (more than one path between two tables)
 - connection trap (an optional path through different entities)
 - fan trap (multiple one-to-many relationships that fan out from a single table)

© 2015 IBM Corporation

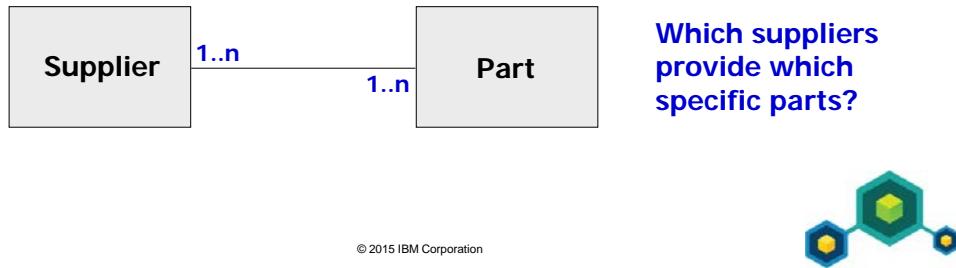


These are data modeling traps, not metadata modeling traps, so you cannot use Framework Manager to fix them in the data source. However it is useful to know of them so that you can make metadata model designs which can handle them and generate the appropriate SQL at run time to provide predictable results. A data trap does not necessarily indicate a problem, only that an area is worth inspection and possible refinement.

The data modeler must carefully examine any area that does not appear to represent the data completely.

Examining Data Traps: Chasm Trap

- Many-to-many relationship
- Structure cannot record and maintain data (it lets the information fall into a chasm)
- Not incorrect when designing at a high level (it just does not show all the necessary details)



In a chasm trap, more than one row in a table is related to more than one row in another table.

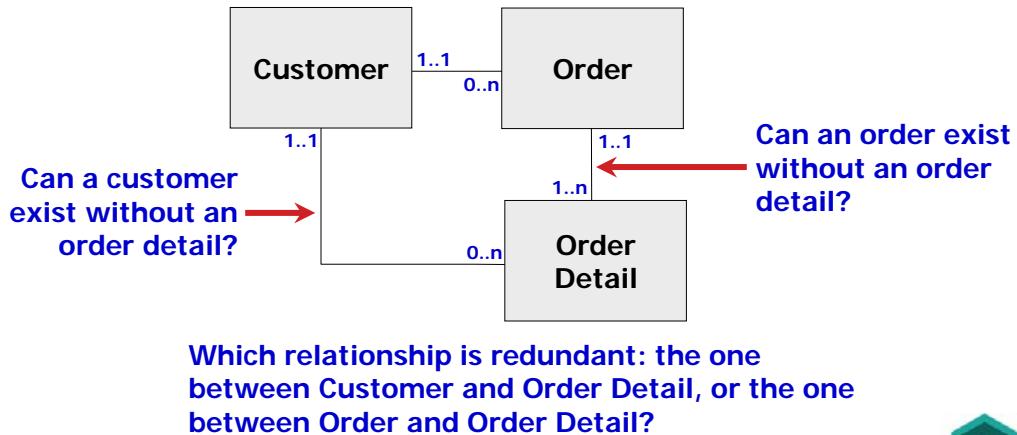
The slide example shows that a single supplier may provide many parts, and many suppliers may provide a single part.

If each supplier can potentially supply every single part, how do you report on the suppliers that provide specific parts?

This is typically resolved with a "bridge" table that records the details of the relationship between the two tables.

Examining Data Traps: Transitive Relationship

- Exists if there is more than one path between two tables



© 2015 IBM Corporation



A transitive relationship trap resembles a wheel shape.

This kind of trap may make it difficult to write queries that retrieve the appropriate data, because it may not be clear which table columns must be included in the query.

Going through either path produces some sort of result, but which path is the correct or more efficient one for your specific query?

Business Analytics software

IBM

Examining Data Traps: Fan Trap

- Identified by multiple one-to-many relationships that fan out from a single table

```

    graph TD
      Division[Division] -- "1..1" --> Branch[Branch]
      Division -- "1..1" --> Employee[Employee]
      Branch
      Employee
      question["Is there a direct relationship between Branch and Employee?"]
      question --> Branch
      question --> Employee
  
```

© 2015 IBM Corporation

Fan traps are also known as parallel relationships.

In this kind of trap, a table has two one-to-many relationships spreading out from it, implying that the two other tables have no connection to each other.

The modeler should examine such traps to determine whether a crucial relationship is missing.

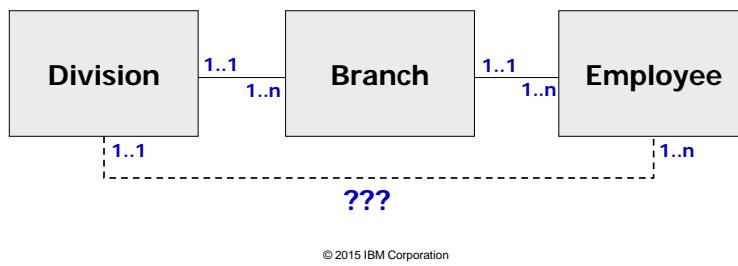
Branch may actually have a direct relationship to Employee.

Examining Data Traps: Connection Trap

- Is there an optional path through different entities?
- There must be a reliable path through all truly related entities

What is the relationship between Branch and Employee?

If an employee does not work for a branch, do they work for a division?

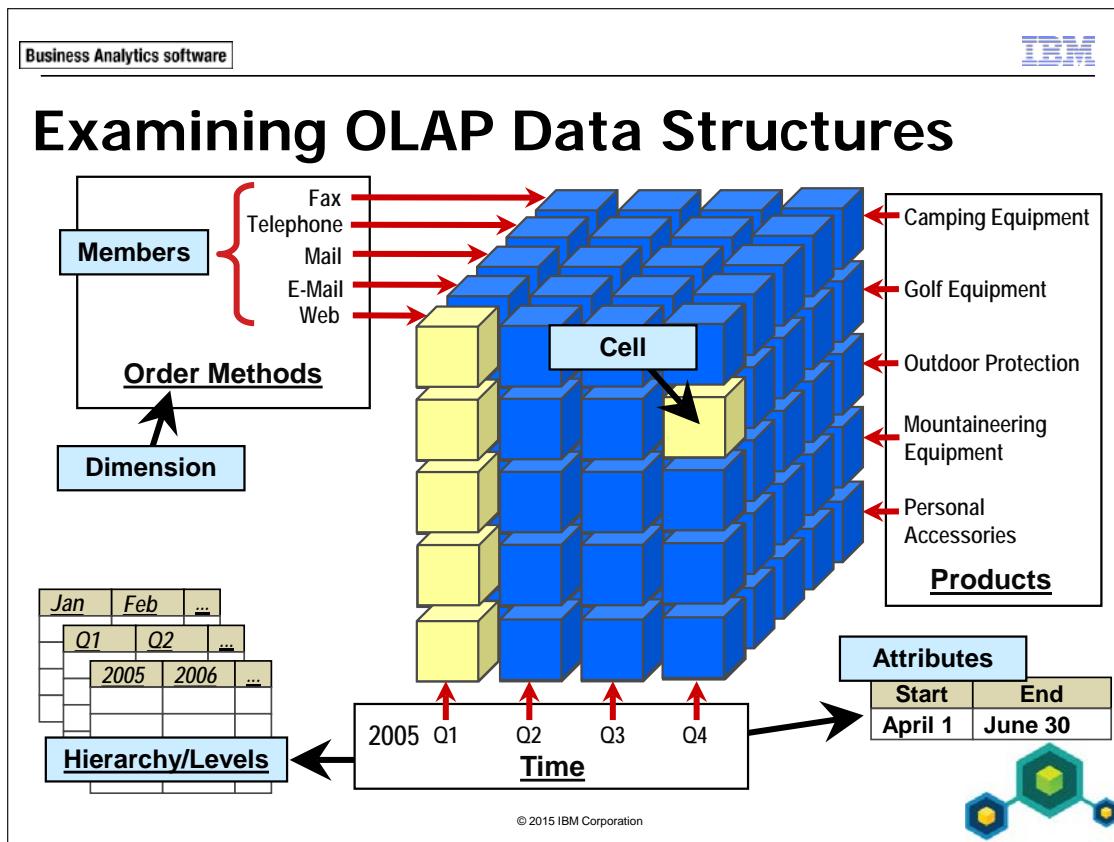


© 2015 IBM Corporation



The slide example suggests that there may be a direct relationship between the Employee and Division tables.

An employee may be able to work directly for a division, instead of working for a branch within a division. For example, the employee may work from a home office.



OLAP is an alternative data access strategy to modeling metadata for Relational models
The OLAP structure consists of the following elements:

- **dimensions** - contain members, which may be structured into hierarchies and levels
- **hierarchies** - provide context to the level structures they contain
- **levels** - provide structure for the members of a hierarchy
- **members** - data entities that provide context to cell values
- **attributes** - provide additional information about members
- **cells** - are intersection points containing values (measures) for various members from different dimensions (also referred to as tuples)

Tools that can be used to build an OLAP data structure include Cognos PowerPlay Transformer and SAP BW. The reporting traps just discussed should already have been modeled out in this kind of structure. The data represents a snap shot in time and must be periodically updated.

Examining OLAP: MOLAP vs ROLAP

- Multidimensional OLAP (MOLAP):
 - transforms data into cubes that are optimized for analytic queries
 - creates multiple copies of the cube, and the ETL process can be lengthy
- Relational OLAP (ROLAP):
 - operates on top of a relational datasource in which data is modeled as star schemas
 - pre-computed summaries provide faster query results

© 2015 IBM Corporation



ROLAP is optimized to work with large data sets. It uses aggregate tables to improve the speed at which large queries are executed.

You can design a ROLAP cube by using the IBM Cognos Dynamic Cube Designer., which is outside the scope of this course.

Identifying Data Access Strategies

- Consider using a cube to analyze data sets that are not prohibitively large.
- Consider creating a star schema to improve performance over an operational system.
- If you need to access live data, you may have no choice but to go with an operational system.

© 2015 IBM Corporation



The methods that you use to access your data for reporting purposes will be determined by your needs and by the structure and content of the data itself.

What are acceptable performance times for your reports? If the information is required quickly (within seconds), consider using cubes. However, if the data set is extremely large, cubes may not be the solution and you may want to consider using a data mart or warehouse with pre-aggregated tables.

If you require live data, you may need to go directly against your operational system. Remember that reporting directly against a normalized operational system may yield poor performance and may slow down the writing of data to the database.

Planning and scope should be given a great deal of attention before embarking on a business intelligence project.

This material is meant for IBM Academic Initiative use only. NOT FOR RESALE

Summary

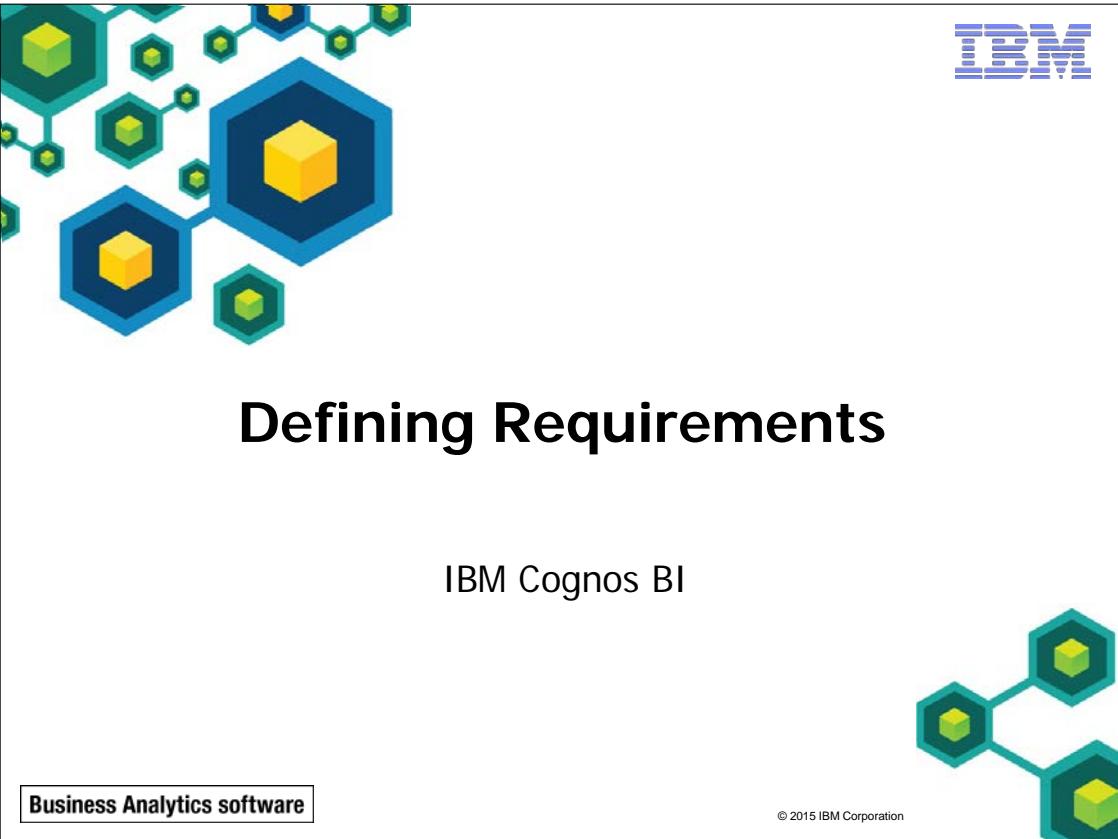
- You should now be able to:
 - define the role of a metadata model in Cognos BI
 - distinguish the characteristics of common data structures
 - understand the relative merits of each model type
 - examine relationships and cardinality
 - identify different data traps
 - identify data access strategies

© 2015 IBM Corporation

This material is meant for IBM Academic Initiative use only. NOT FOR RESALE

© 2003, 2015, IBM Corporation

This guide contains proprietary information which is protected by copyright. No part of this document may be photocopied, reproduced, or translated into another language without a legal license agreement from IBM Corporation.



This material is meant for IBM Academic Initiative use only. NOT FOR RESALE

© 2003, 2015, IBM Corporation

This guide contains proprietary information which is protected by copyright. No part of this document may be photocopied, reproduced, or translated into another language without a legal license agreement from IBM Corporation.

Objectives

- At the end of this module, you should be able to:
 - examine key modeling recommendations
 - define reporting requirements
 - explore data sources to identify data access strategies
 - identify the advantages of modeling metadata as a star schema
 - model in layers

© 2015 IBM Corporation

This material is meant for IBM Academic Initiative use only. NOT FOR RESALE

© 2003, 2015, IBM Corporation

This guide contains proprietary information which is protected by copyright. No part of this document may be photocopied, reproduced, or translated into another language without a legal license agreement from IBM Corporation.

3-3

Modeling Recommendations (1 of 2)

1. Define reporting requirements and data access strategies.
2. Import only required reporting objects in a phased approach and alter as little as possible.
3. Verify relationships and query item properties.
4. Model in freehand to identify query usage.
5. Use model query subjects to control query generation and usage and to consolidate metadata.

© 2015 IBM Corporation



1. Before beginning any modeling exercises, determine what the reporting requirements are. This will help you to find the correct data and define a data access strategy.
Based on available data sources, data volumes, and environmental factors such as network speed, hardware processing power, and so on, an appropriate data access strategy should be planned and implemented to ensure acceptable response times to report requests.
2. Import only what is required for reporting and import it in manageable chunks. Alter data source query subjects as little as possible. Leaving the data source query subjects as simple all-inclusive select statements reduces future maintenance. For example, when a table has a new column added to it, simply update the data source query subject that references it in Framework Manager and the new column will appear as a new query item.

This material is meant for IBM Academic Initiative use only. NOT FOR RESALE

3. Verify that imported relationships reflect those in the data source and that the query item properties are set correctly. For relationships, notice:
 - cases where a dimension query subject relates to a fact query subject on different keys
 - cases where there are multiple valid relationships between query subjects
 - dimension query subjects that belong to multiple hierarchies
4. Model in freehand to identify modeling challenges and how query subjects are used (which query subjects are treated as facts, dimensions, or both). Identifying these issues on paper can provide a clear modeling plan.
5. Begin creating simplified, abstracted model query subjects to resolve modeling challenges by:
 - creating aliases where required to control query paths
 - modeling as a virtual star schema to control SQL generation (what is a fact, what is a dimension)
 - removing descriptive (dimensional) attributes from fact tables
 - consolidating related information into one model query subject for a cleaner presentation (for example, placing all product related query items in one model query subject)

Modeling Recommendations (2 of 2)

6. Customize metadata for run time.
7. Specify determinants as required.
8. Resolve multiple ambiguous joins between query subjects.
9. Create analysis objects if OLAP-style querying is required.
10. Create the business view as a set of star schema groupings.

© 2015 IBM Corporation



6. Customize metadata for runtime by using:
 - parameter maps and session parameters to handle dynamic column or row retrieval
 - prompt values and query macros to add mandatory user prompts and security filters
7. Specify determinant information where required to enable accurate aggregation in cases where a level of granularity has repeating keys, your data contains BLOBs, or you wish to avoid the distinct clause on unique values when grouping or enhance performance for regular dimensions.
8. Resolve any relationship ambiguities, such as multiple joins between two query subjects, by deleting surplus joins and by creating role-playing dimensions.
9. Create regular and measure dimensions if authors need to perform OLAP-style queries on relational data.
10. Use star schema groupings to build logical business groupings in the business view and to indicate conformed dimensions based on naming conventions.

These recommendations apply to both operational and dimensional (star schema) data sources. If proper design has been implemented, then dimensional data will usually require much less modeling in the Framework Manager environment.

These recommendations are designed to be a guideline. Modelers must do what is appropriate for their situation. These recommendations cannot account for every modeling need, and therefore, modelers will always need to decide when to use some or all of these recommendations, and when to model outside of the paradigm presented here.

Analyzing Data Requirements

- identify the business-intelligence-related problems to be solved
- consider your data access strategy:
 - reporting versus operational
 - metadata
 - multiple data sources
 - modeling requirements
 - relationships

© 2015 IBM Corporation



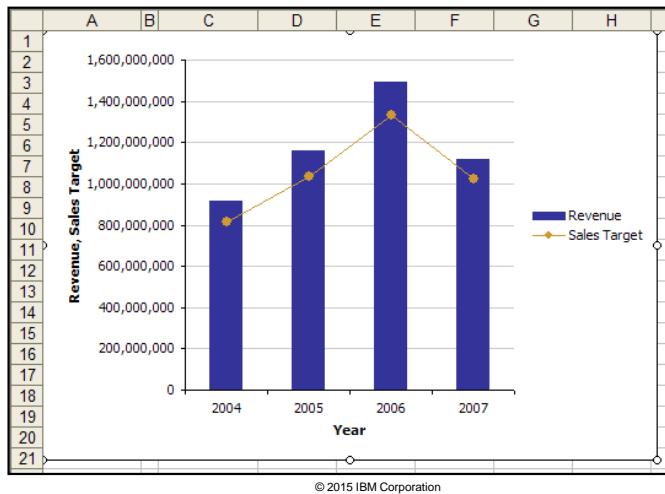
Problems to be solved include setting the scope of your project and setting its methodology, as well as multilingualism, performance, security, and presentation. You should ask questions such as:

- Do you and the IBM Cognos users agree on the model requirements?
- Does the data source contain the data and metadata you need?
- Does the same data exist in more than one source?
- Which data source tables are the fact tables, which are the dimensions, and which are both fact tables and dimensions? What are the keys and attributes of each dimension?
- Do fact tables contain only facts and foreign keys? Do they also contain dimensional attributes that should be in dimension tables?
- What are the required relationships and are there multiple paths between tables?

This material is meant for IBM Academic Initiative use only. NOT FOR RESALE

Gathering Requirements

- interview authors and users to determine needs
- view reports that IBM Cognos will be replacing



Ralph Kimball (author on the subject of data warehousing and business intelligence) recommends an interview-based approach to determine report requirements. The focus is on what information the key business decision makers require to do their jobs. This will result in a framework of key performance indicators (KPI) and business contexts.

Overview of the Sample Outdoors Company

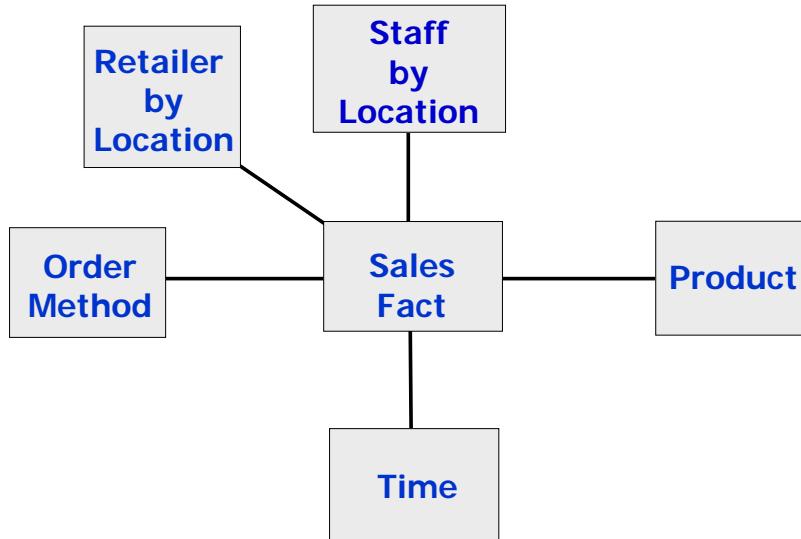
- the Samples Outdoors Company operational relational database is called GOSALES
- use GOSALES to model metadata for various scenarios
- focus on a portion of this database to develop a sales reporting model

© 2015 IBM Corporation



Although a reporting (star schema) database is recommended for performance and ease of use, in this course you will model an operational system to provide you with a wider variety of modeling techniques and prepare you for more data scenarios back at your place of work.

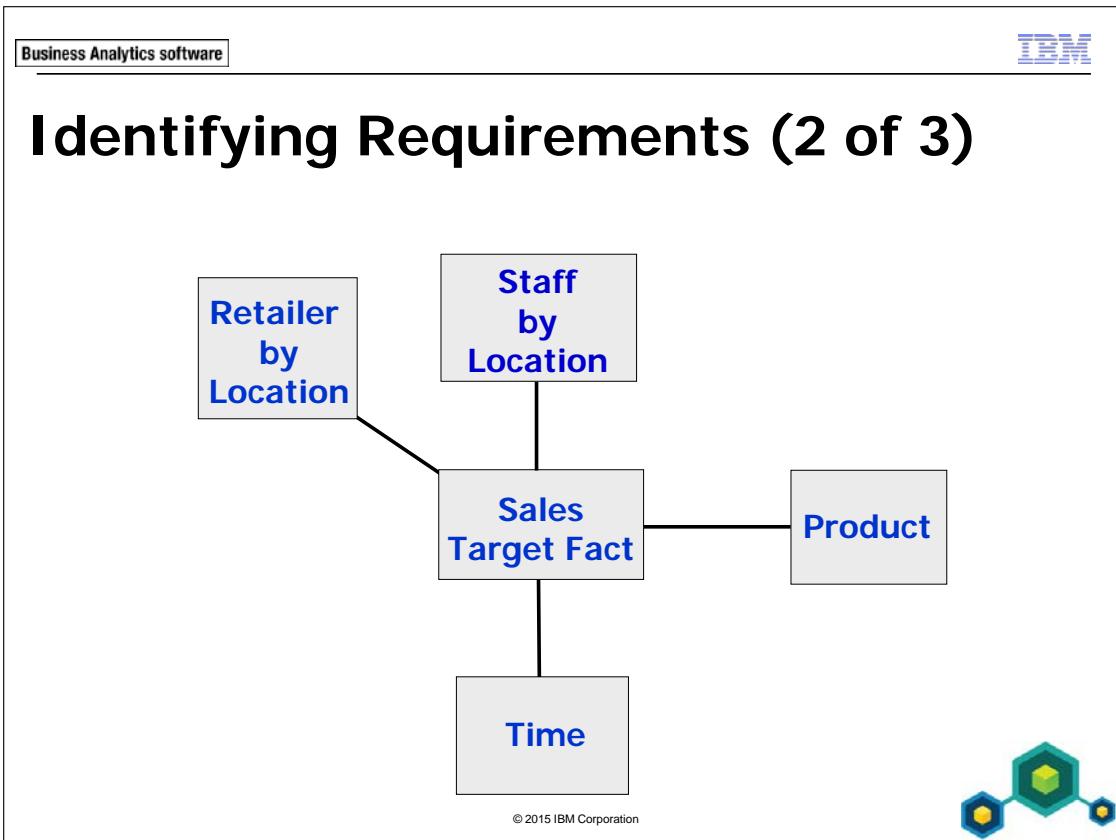
Identifying Requirements (1 of 3)



Before creating a Framework Manager project, you must:

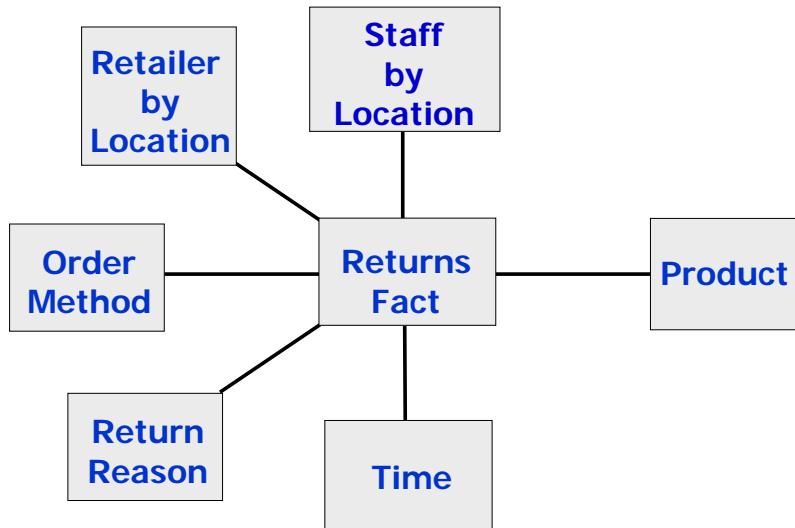
- analyze and understand your reporting requirements based on the business process that authors will be reporting on
- understand your data and the structure of your data source(s)

Upon interviewing BI users, you discover that they would like to report on sales data by the dimensions shown in the slide example above.



BI users would also like to report on sales targets by the dimensions shown above.

Identifying Requirements (3 of 3)



© 2015 IBM Corporation



And finally, BI users would like to report on returns by the dimensions shown above.

Knowing this information, you interview the database administrator to find out where this information can be obtained. You are told it is in the GOSALES database.

The B5A52-Requirements_Handout.doc file (in C:\Edcognos\B5A52\Instructor Files) identifies the dimensions that relate to each fact in the GOSALES database, along with diagrams that give a visual representation of these relationships in a star schema format. This table includes more dimensions than are presented in this module, since additional logical dimensions will be discovered throughout the modeling process. It will be a useful reference as you work their way through the course.

Demo 1: Explore the Data Source

Purpose:

As the modeler for The Sample Outdoors Company, your task is to develop a model that supports the business requirements of report and ad hoc query authors working with Cognos BI. Before you begin the modeling process, you will locate the data sources identified as necessary to meet those reporting requirements.

Component: **IBM Data Studio**

Task 1. Open IBM Data Studio.

1. From the **Start** menu, point to **All Programs > IBM Data Studio**, and then click **Data Studio 4.1.0.0 Client**.
2. Click **OK** to close the **Workspace Launcher**.
If the connection has not been configured, perform steps 3-5, otherwise continue with step 6.
3. From the **Administrator Explorer** pane, expand **localhost>50000** and then double-click **GS_DB [DB2 Alias]**.
4. In the **Properties for GS_DB** window, on the **General** tab, in the **User name** field type **db2admin**, and then in the **Password** field type **Education1**.
5. Click the **Save password** check box, and then click **OK**.
6. Expand **All Databases > localhost > DB2**, right-click **GS_DB**, and then click **Connect**.
7. Click the **Tables** folder.

Task 2. Explore the Tables.

The sales reports will be focused on orders, so note the ORDER_DETAILS, ORDER_HEADER, and ORDER_METHOD tables.

1. Double-click **ORDER_HEADER** to view details for the ORDER_HEADER table in the Details pane.

You can identify the tables related to the ORDER_HEADER header table by looking for foreign keys. Here you can see several keys: ORDER_NUMBER, RETAILER_SITE_CODE, RETAILER_CONTACT_CODE, SALES_STAFF_CODE, SALES_BRANCH_CODE, and ORDER_METHOD_CODE. You will now examine some of these relationships.

2. Click the **Back** arrow, and then double-click **ORDER_DETAILS** to view details for the ORDER_DETAILS table in the Details pane.

You can see the facts needed for many reports: QUANTITY, UNIT_COST, UNIT_PRICE, and UNIT_SALE_PRICE. You also see ORDER_NUMBER, confirming the relationship to ORDER_DETAILS. You also see PRODUCT_NUMBER. You will confirm that this is the link to the Product data.

3. Click the **Back** arrow.

You can see a series of tables whose names start with PRODUCT_, so you will have to view each one to decide if they contain data that meets the reporting requirements.

4. Double-click **PRODUCT** to view details for the PRODUCT table in the Details pane.

You see PRODUCT_NUMBER, confirming that this is the link to the sales (ORDER_DETAIL) data. Upon asking the Database Administrator about BASE_PRODUCT_NUMBER, you learn that this is for handling multiple brands of a given product. You see several codes, indicating foreign keys to PRODUCT_TYPE, PRODUCT_COLOR, PRODUCT_SIZE, and PRODUCT_BRAND. However you don't see a product name or description field. You will have to keep looking.

5. Click **Back**, and then double-click **PRODUCT_TYPE** to view details for the PRODUCT_TYPE table in the Details pane.

You see PRODUCT_TYPE_CODE, confirming the link to PRODUCT. You also see PRODUCT_LINE_CODE (which you can confirm links to PRODUCT_LINE in a three-level hierarchy; product line>product type>product).

Note the many PRODUCT_TYPE_xx columns. This is one form of multilingual support, a column for each language. Later in the course, you will see how you can modify the metadata so that authors and consumers can access a single product type column, which will automatically return the appropriate language based on the user's regional settings. Recall that you did not see such a list of columns for the name of the product itself in PRODUCT, so that table must have a different form of multilingual support.

6. Click **Back**, and then double-click **PRODUCT_NAME_LOOKUP** to view details for the PRODUCT_NAME_LOOKUP table in the Details pane.

Now you see PRODUCT_NAME and PRODUCT_DESCRIPTION as well as PRODUCT_LANGUAGE, and you can conclude that this table must have multiple language records for each PRODUCT_NUMBER.

The only outstanding report item required before beginning a baseline project is an order method. Recall that ORDER_HEADER holds an ORDER_METHOD_CODE column.

7. Click **Back**, and then double-click **ORDER_METHOD**.

You see the order method columns; one for each supported language in the data. You will need to resolve this multilingual challenge in the same manner you will use for PRODUCT_TYPE_xx.

8. Close **IBM Data Studio**.

Results:

You have identified the following tables as those needed for the first phase of your model building, a baseline project.

- ORDER_HEADER, ORDER_DETAILS, ORDER_METHOD
- PRODUCT, PRODUCT_TYPE, PRODUCT_LINE,
- PRODUCT_NAME_LOOKUP**

Best Practice: Modeling in Stages

- a common tendency is to immediately import all metadata to meet all reporting requirements
 - this can create a complex set of objects to work with as a starting point
- recommended approach:
 - develop model in stages; start with a subset of report requirements
 - import additional metadata as needed
 - revisit and revise as required (iterative development)

© 2015 IBM Corporation

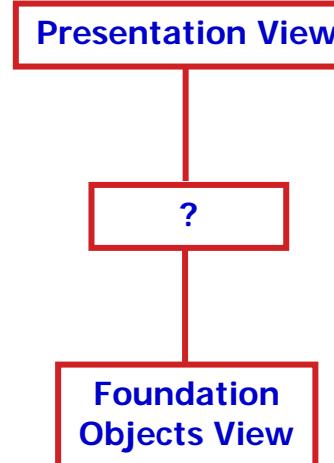


For your course model, the scenario is that you have interviewed functional users and have determined that you will start with a baseline project that allows reporting on sales by product or by order method. This postpones work on sales staff, location, retailer (your customers), sales targets, and returns.

As you progress in your modeling activities, you may discover other requirements or the data itself may present you with other reporting options that were not thought of by the end users.

Modeling in Layers

- we recommend a Presentation View for logical groupings
- choose whether you would like a middle layer, and what you want to put in it
- metadata is imported into the lowest layer



© 2015 IBM Corporation



A Presentation View contains only the star schema groupings. This logically groups objects appropriate for the business and easily allows you to create separate packages for different reporting needs.

The decision to have another layer between the Foundation Objects View and the Presentation View involves several factors. For example, what is the size of the model? Do you need to reduce development time rather than ensure ease of maintenance later in the modeling cycle?

The next three slides investigate some approaches.

Modeling in Layers: No Middle Layer

- Foundation Objects View contains:
 - data source query subjects
 - model query subjects
 - calculations and filters
 - appropriate for large projects
 - difficult to maintain if data source structure changes frequently
 - not easily portable between different types of databases

Presentation View

Foundation Objects View



© 2015 IBM Corporation

This method requires the least duplication of query subjects, keeping the physical size of the project files to a minimum. It is best suited to large implementations or situations where a data warehouse has already been set up to accommodate the majority of the specialized business logic for reporting.

While it requires less development time, this method can require more maintenance when in production, since you will need to remodel to reflect any changes to the underlying data source objects, since the published objects are simply shortcuts to the data source query subjects. Therefore, if you expect ongoing changes to the underlying data structure, this may not be a suitable option.

You can view this approach in the GO Operational - (No Middle Layer) model located at C:\Edcognos\B5A52\CBIFM-Files\Alternate_Models.

Business Analytics software

IBM

Modeling in Layers: Business Logic View

- Business Logic View contains
 - all model query subjects, joined with relationships
 - all calculations and filters
- provides insulation between data source and reports
- requires less maintenance
- portable
- longer development time
- larger project files and model

```

graph TD
    PV[Presentation View] --- BLV[Business Logic View]
    BLV --- DSV[Data Source View]
  
```

© 2015 IBM Corporation

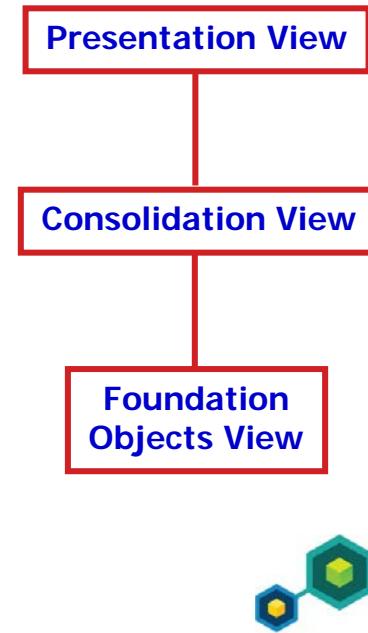
Using a business logic layer lets you set up the complex queries and reuse foundation layer objects in multiple locations. This provides insulation from the underlying data source for the reports. No work is required in the Data Source View since it is all done in the Business Logic View.

Creating the model query subjects (and rebuilding all their relationships) in the Business Logic View takes extra work, but it provides a layered structure for improved model maintenance, readability and portability. For example, to move the application from one database vendor to another, all you have to do is re-map the model query subjects in the Business Logic View to the tables in the new data source. There is no need to rebuild the reports, or remodel the metadata.

You can view this approach with the GO Operational - (Business Logic View) model located at C:\Edcognos\B5A52\CBIFM-Files\Alternate_Models folder. Model query subjects in this view are also used to consolidate hierarchy items for presentation (no relationships). This can be seen in the folders of the Business Logic View (example: Product in the Product Info folder).

Modeling in Layers: Consolidation View

- model query subjects, calculations and filters are split between Consolidation and Foundation Objects Views, based on need
- all relationships are in the Foundation Objects View
- compromise between development and maintenance time
- smaller project size than Business Logic View method

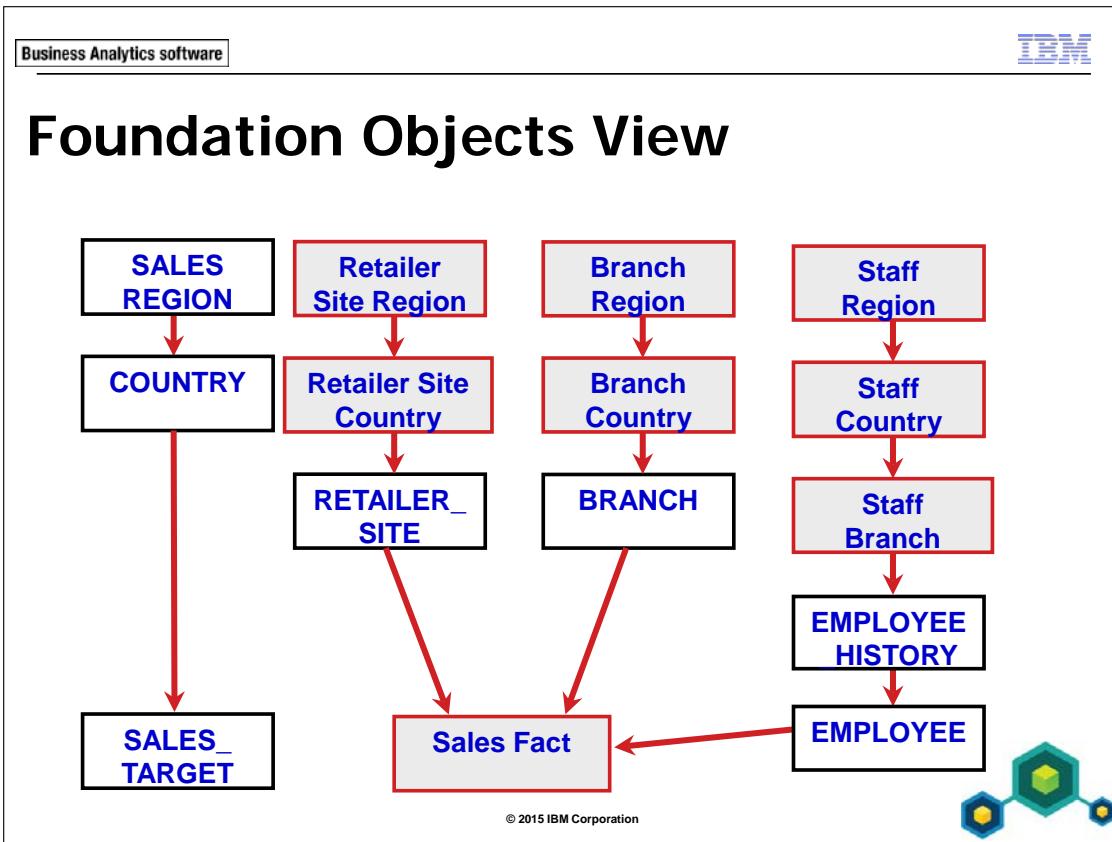


© 2015 IBM Corporation

In this compromise between the previous two methods, you create model query subjects and their relationships in the lowest layer. The middle layer acts as a consolidation layer with some business logic (where required). For example, this view is where snowflake dimensions are consolidated. It also acts as an insulation layer between reports and the data source.

Calculations may appear in either the lowest or the middle layer, depending on where the related model query subject is created. A model query subject that is created to resolve a reporting issue will be created in the lowest layer. Therefore, any calculations required for that object will be defined in the lowest layer.

The Business Logic View and Consolidation View methods create additional model query subjects even when data source query subjects present no issue and could technically be used in the Presentation View. This can affect physical file size noticeably. The Consolidation View method is the middle ground and is the approach that this course will use.



This course will use the Consolidation View approach in the demos and workshops. You can view the results of this approach in the CBIFM-Final model located at C:\Edcognos\B5A52\CBIFM-Files\Final.

After the Foundation Objects View layer is completed, it will contain:

- all data source query subjects
- any model query subjects required to resolve reporting issues
- some calculations and filters where appropriate

The above diagram identifies, in the grey boxes (with mixed-case text), the model query subjects used to resolve reporting issues. Sales Fact acts as a view of ORDER_HEADER and ORDER_DETAILS, thereby controlling the SQL generation. All others model query subjects are used to create aliases to control query paths.

Summary

- You should now be able to:
 - examine key modeling recommendations
 - define reporting requirements
 - explore data sources to identify data access strategies
 - identify the advantages of modeling metadata as a star schema
 - model in layers

© 2015 IBM Corporation

This material is meant for IBM Academic Initiative use only. NOT FOR RESALE

© 2003, 2015, IBM Corporation

This guide contains proprietary information which is protected by copyright. No part of this document may be photocopied, reproduced, or translated into another language without a legal license agreement from IBM Corporation.



The advertisement features a white background with a decorative pattern of teal and yellow hexagonal icons. A large, central teal hexagon contains a yellow 3D cube. The IBM logo is positioned in the top right corner. Below the pattern, the title "Creating a Baseline Project" is displayed in a large, bold, black sans-serif font. In the center, the text "IBM Cognos BI" is written in a smaller, black sans-serif font. At the bottom left, a small rectangular box contains the text "Business Analytics software". At the bottom right, there is a small copyright notice: "© 2015 IBM Corporation".

Creating a Baseline Project

IBM Cognos BI

Business Analytics software

© 2015 IBM Corporation

This material is meant for IBM Academic Initiative use only. NOT FOR RESALE

© 2003, 2015, IBM Corporation

This guide contains proprietary information which is protected by copyright. No part of this document may be photocopied, reproduced, or translated into another language without a legal license agreement from IBM Corporation.

Objectives

- At the end of this module, you should be able to:
 - follow the IBM Cognos and Framework Manager workflow processes
 - define a project and its structure
 - describe the Framework Manager environment
 - create a baseline project
 - enhance the model with additional metadata

© 2015 IBM Corporation

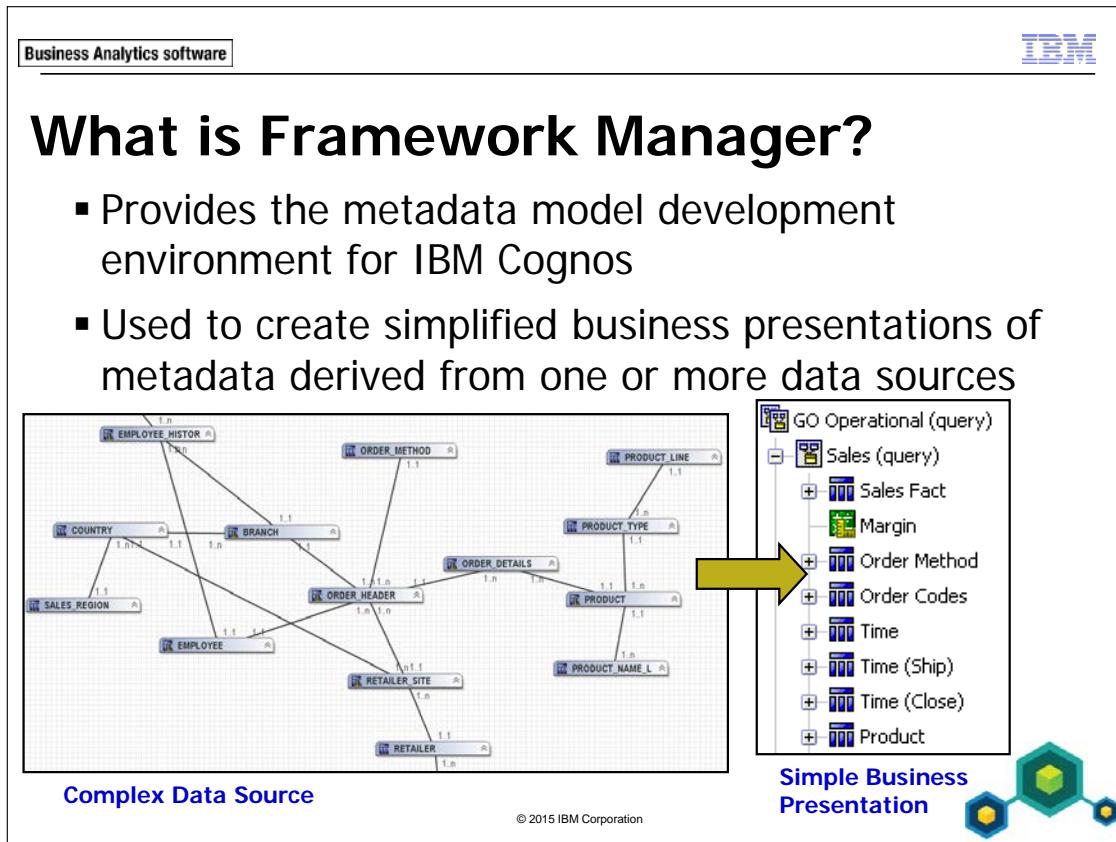
Unless otherwise specified in demo or workshop steps, you will always log on to IBM Cognos in Local LDAP namespace using the following credentials:

- User ID: admin
- Password: Education1

This material is meant for IBM Academic Initiative use only. NOT FOR RESALE

© 2003, 2015, IBM Corporation

This guide contains proprietary information which is protected by copyright. No part of this document may be photocopied, reproduced, or translated into another language without a legal license agreement from IBM Corporation.



When you work in Framework Manager, you work in a project. A project contains metadata objects (the model) organized for report authors according to the business model and rules. A model in Framework Manager is a business presentation of the structure of the data from one or more data sources. A model defines the metadata objects, structure, and grouping, as well as relationships and security.

The modeler's job is to take the complexity of the underlying data structures and create simplified presentations for authors and business analysts that provide predictable results. These simplified views are then published as packages to the IBM Cognos Connection for use in the BI studios.

Framework Manager is a Windows-based client application as opposed to Business Workspace, Query Studio, Report Studio, and Cognos Connection, which are Web-based applications.

Business Analytics software

IBM

How does Framework Manager Connect to IBM Cognos BI?

- The connection between Framework Manager and the Cognos BI dispatcher is defined in Cognos Configuration

© 2015 IBM Corporation

When installing Framework Manager on a different computer than other IBM Cognos BI components, the installer must define a connection that allows Framework Manager to communicate with the dispatcher, either directly or through a dedicated gateway. This connection allows the modeler to publish packages to the Cognos BI environment, by ensuring that Framework Manager connects to the same URI as the non-modeling components of IBM Cognos BI.

Framework Manager Query Modes

- two query modes: compatible or dynamic
- dynamic query mode (DQM) is best for new IBM Cognos BI applications that have supported data sources
- DQM optimizes query planning, execution, and results in supported environments
- projects created in version 10.1 (or earlier) are automatically in compatible query mode (CQM)
- existing CQM projects can be tested, published, or changed to dynamic query mode

© 2015 IBM Corporation



When you create a project in Framework Manager 10.2, you can select which query mode it will use. If your data sources are supported, DQM is the recommended mode, as it improves query performance by reducing query complexity and using in-memory caching to optimize query execution.

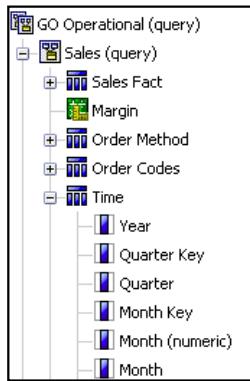
Projects created in Framework Manager 10.1 (or earlier versions) are automatically in CQM. However, you can still test model objects and publish packages in DQM, or even change the Query Mode property to dynamic query mode. Note that, before you change the mode in which you publish a package, you should use IBM Lifecycle Manager to identify the impact on any reports that use that package.

This course uses CQM-enabled models, due to the restrictions of the classroom environment.

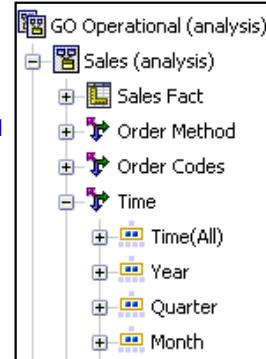
Framework Manager Model Types

- You can create two model types:
 - relational for reporting
 - dimensionally modeled relational (DMR) for OLAP-style analysis and reporting

Relational Model



Dimensional Model



© 2015 IBM Corporation



Relational models consist of query subjects and within those, query items.

DMR models consist of dimensional information provided by the modeler, such as hierarchies and levels, to allow authors to perform OLAP-style queries.

These two model types can be developed in the same project and published as separate packages. In the slide example, the packages are GO Operational (query) and GO Operational (analysis), both published from the same project.

Business Analytics software

IBM

Framework Manager Projects

- at the highest level, objects in a project include:
 - a model organized into namespaces and folders
 - data sources
 - parameter maps
 - packages

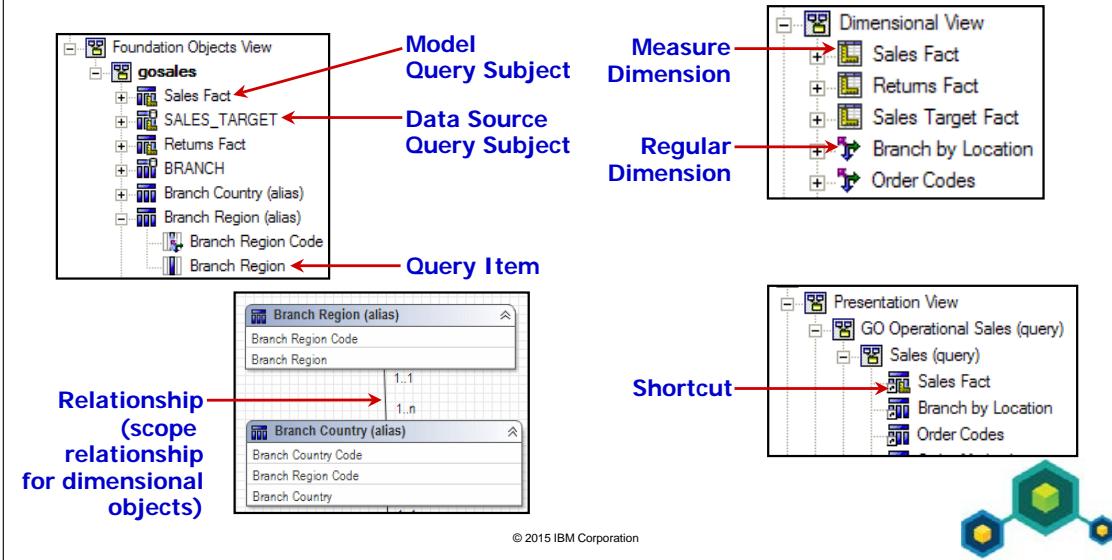
© 2015 IBM Corporation

The system files of a project consist of a folder containing a project file (.cpf) and a set of XML files specific to that project. These files should be backed up or checked into a repository control system regularly to prevent data loss. A project contains the following objects:

- Model - the set of metadata objects organized and modeled for report authors. The root of the model is a namespace that can contain other namespaces and folders, which are containers for organizing objects; however the names of objects in a namespace are qualified with the namespace name to uniquely identify them. This allows you to have objects of the same name in different namespaces.
- Data sources - information IBM Cognos uses to connect to data sources
- parameter maps (optional) - a list allowing the substitution of one value for another at run time
- Package - a subset of the project objects that is published to IBM Cognos Connection for use by authors and business analysts.

Defining Metadata Elements

- Metadata is organized into the following elements:

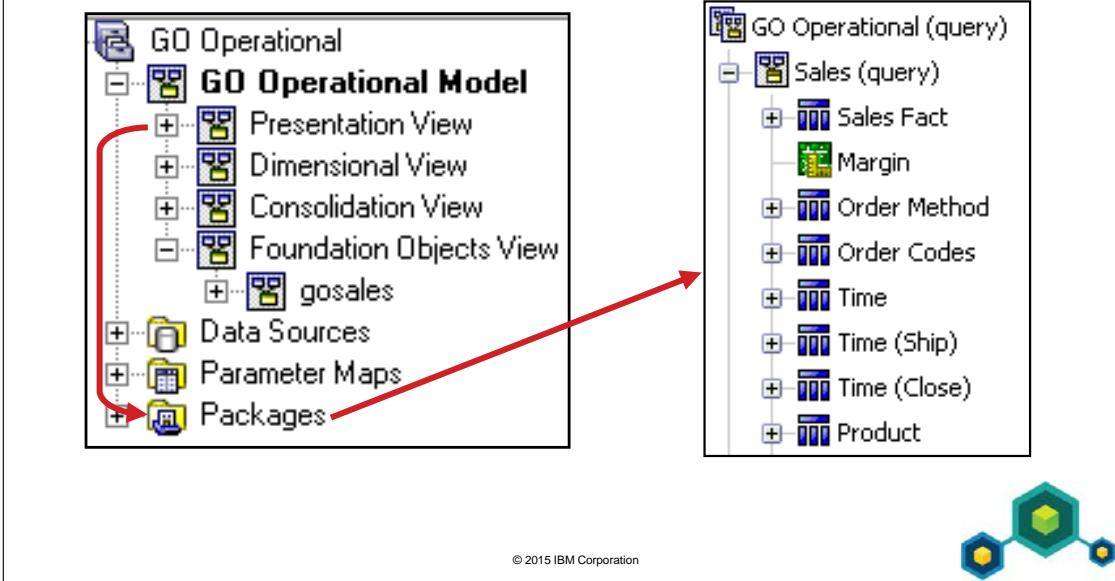


In Framework Manager, you will interact with the following objects:

- **Query subjects** - map to objects in the data source, such as tables, views, synonyms, procedures, or functions. There are three types of query subjects:
 - **Data source**: maps to a corresponding object in the data source and uses a modifiable SQL statement to retrieve the data.
 - **Model**: maps to existing metadata in the model.
 - **Stored procedure**: executes a database stored procedure to retrieve or update the data.

- **Query items** - contained within a query subject and maps to a corresponding object in the data source. Query items are contained in query subjects. For example, a query subject that references an entire table contains query items that represent each column in the table.
- **Relationships** - represent the connection that explains how the data in one query subject relates to the data in another. When you create a relationship, you define the cardinality of each end of the relationship.
- **Regular dimensions** - contain descriptive and business key information, and organize the information in a hierarchy from the highest level of granularity to the lowest, allowing for OLAP-style queries. Regular and measure dimensions map to query items. You would work with regular and measure dimensions when modeling dimensionally. You model dimensionally when your report and business analysts require OLAP-style queries, reports, and analyses.
- **Measure dimensions** - a collection of facts for OLAP-style queries.
- **Scope relationships** - exists between measure dimensions and regular dimensions to define the level at which the measures are available for reporting.
- **Shortcuts** - pointer to an underlying object that can act as an alias or reference. In this course we mainly use shortcuts to create star schema groupings for presentation of logical groupings to the authors.

What do Report Authors View?



Within IBM Cognos Connection and the various studios, report authors interact with a run-time version of a subset of the Framework Manager model. This subset, which the modeler publishes as a package from Framework Manager to the IBM Cognos server, contains metadata that exists in the development model. It is packaged in a structure reflecting a viewpoint appropriate for report authors and business analysts. The actual data is not published with the package; it is retrieved from the underlying data source at run time.

In Framework Manager, you can create several packages containing different subsets of the model.

Demo 1: Examine the Final Project

Purpose:

Before you begin the modeling process for your Sample Outdoors model, you will explore the Framework Manager UI and view the final results of the model you will build by the end of this course. This will help bring context to the tasks ahead of you.

Components: Framework Manager, IBM Cognos Workspace Advanced

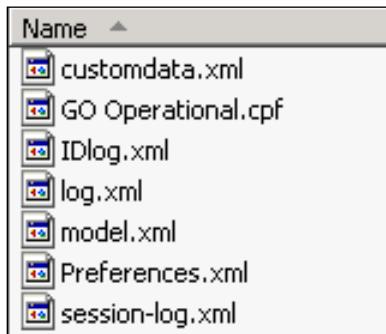
Project: GO Operational

Package: GO Operational (query)

Task 1. Examine the Framework Manager UI and view the top-level objects.

1. Launch Windows Explorer, and then navigate to **C:\Edcognos\B5A52\CBIFM-Files\Final**.

The results appear as follows:

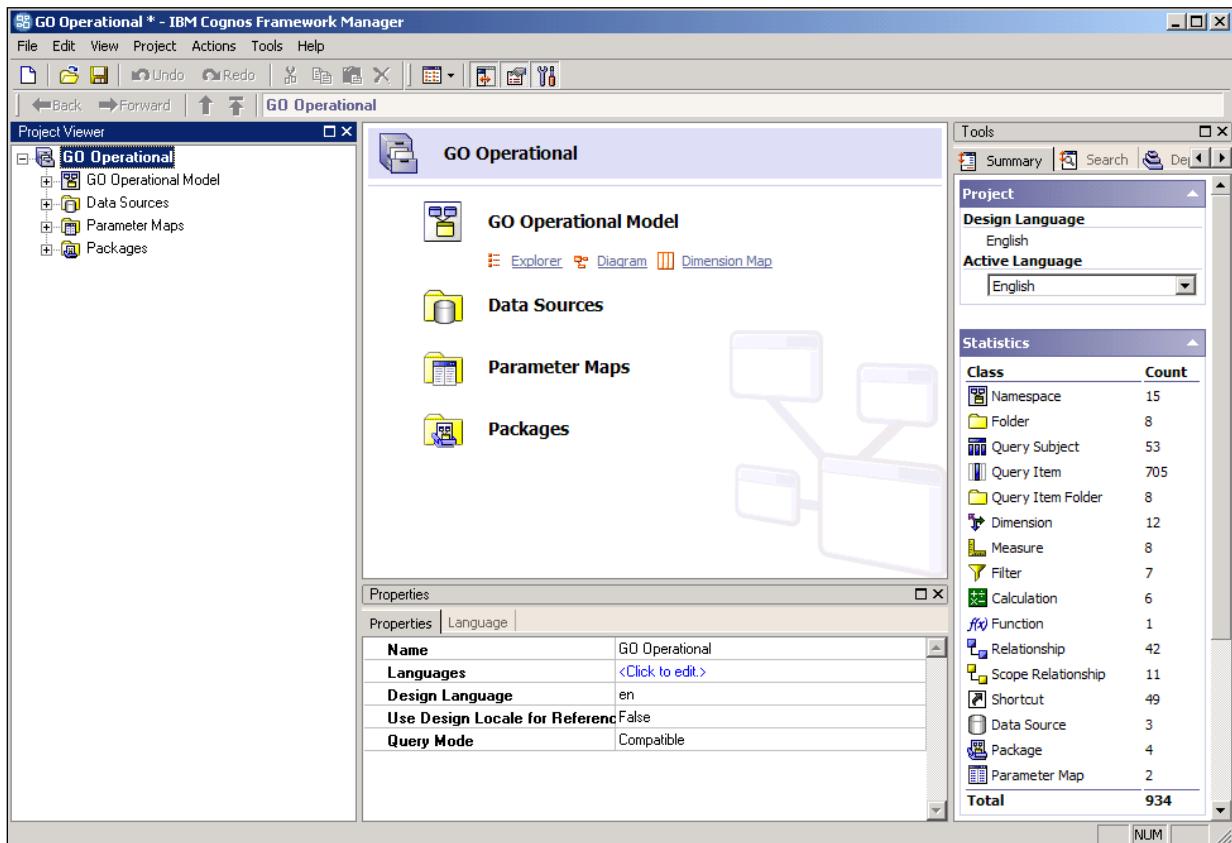


Here you see the project files for the final Framework Manager model you will be building throughout this course. The cpf file is the main project file. The xml files store model information, preferences, session logging, action logging (each modeling action you make), and so on.

2. From the **Start** menu, point to **All Programs>IBM Cognos 10**, and then click **IBM Cognos Framework Manager**.
3. Click **Open a project**, navigate to **C:\Edcognos\B5A52\CBIFM-Files\Final**, and then double-click **GO Operational cpf**.

4. Log in as **admin/Education1**.

The results appear as follows:



Take the time to examine the different UI elements.

- **Project Viewer pane:** provides access to all of your project's objects in a tree format.
- **Project Info pane:** provides access to the project's objects through three tabs (Explorer, Diagram, and Dimension Map) that allow you to create, edit, configure, or delete objects.
- **Properties pane:** allows you to configure various properties for any of the project's objects.
- **Tools pane:** lets you quickly switch the project language, view project statistics, and perform common tasks for selected objects. This pane also provides a search utility (second tab) and an object dependencies utility (third tab). Simply drag an object (and its children if it has any) to the top panel, select the object or one of its children in the top panel and view the dependent objects in the bottom panel. This is very useful when you want to change an object and assess the impact that the change will have on other objects in the model.

This material is meant for IBM Academic Initiative use only. NOT FOR RESALE

© 2003, 2015, IBM Corporation

This guide contains proprietary information which is protected by copyright. No part of this document may be photocopied, reproduced, or translated into another language without a legal license agreement from IBM Corporation.

All panes can be hidden except the Project Info pane, which is the main work area. To bring them back, simply select them from the View menu or use the toggles on the toolbar. You can also detach and rearrange the Project Viewer, Properties and Tools panes.

5. Expand **Data Sources**, click **GOSALES**, and then expand **Type** in the **Properties** pane.

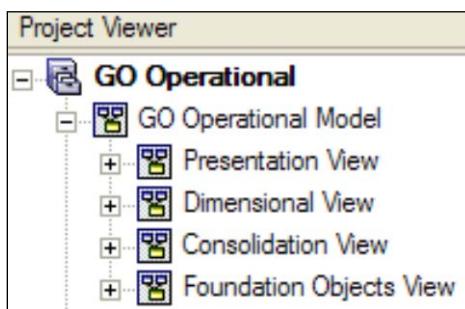
The results appear as follows:

Properties	Language
Content Manager Data Source	GOSALES
Catalog	
Cube	
Schema	GOSALES
Type	
Query Type	Relational
Interface	OL
Function Set ID	<Click to edit>

Notice the Query Type property indicates that these are relational sources.

6. In the **Project Explorer**, expand **GO Operational Model**.

The results appear as follows:



Notice the four namespaces for logical separation of objects by function. These are the naming conventions used in this course. You can use whatever naming convention and object organization you are comfortable with.

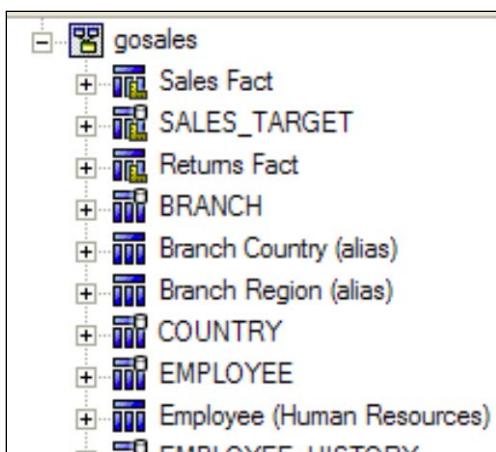
- **Consolidation View** is an isolation layer. It is made up of model query subjects that are used as containers to consolidate multiple query subjects into one (such as Product, Product Type, and Product Line). It also simplifies query subjects by hiding codes or organizing them in subfolders. This layer insulates reports from potential changes in the underlying data source.

- **Dimensional View** contains multidimensional objects based on model query subjects in the Consolidation View. These are used for OLAP-style queries and are based on objects in the Consolidation View.
- **Presentation View** contains groupings of shortcuts that present the underlying objects in a logical presentation.
- **Foundation Objects View** contains the "base" query subjects; data source query subjects as well as any model query subjects used to resolve reporting issues.

Task 2. View the detail objects.

1. Expand **Foundation Objects View > gosales**.

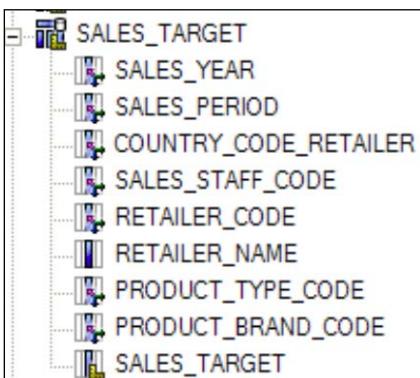
The results appear as follows:



The gosales namespace contains all the data source query subjects (represented with a small database icon) created during the import process from the GOSALES database as well as model query subjects that are used to alias original data source query subjects or to create views (merge query subjects together to create "As View" behavior) of original data source query subjects. The small yellow angle ruler icon indicates the query subject contains measures as seen on Sales Fact, SALES_TARGET, and Returns Fact.

2. Expand **SALES_TARGET**.

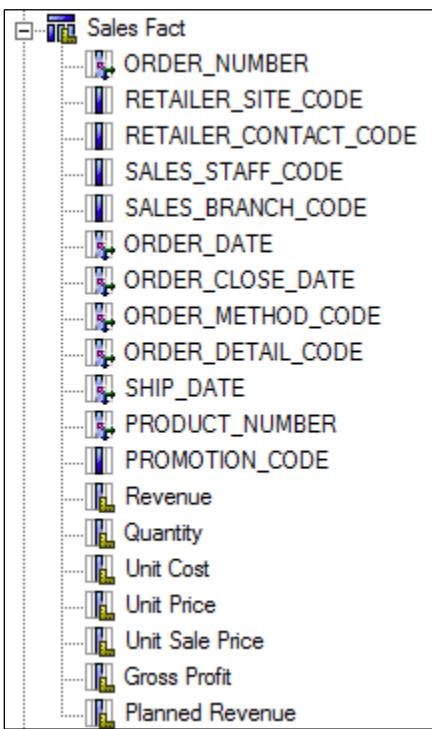
The results appear as follows:



Notice that, with the exception of a few property changes, these data source query subjects are untouched after being imported. This is recommended to ease maintenance.

3. Expand **Sales Fact**.

The results appear as follows:



This is a model query subject that is the result of merging two data source query subjects together to create "As View" behavior.

Query items to be used by authors have been given user-friendly names. This is not needed for query items that will not be included in the final Presentation View. Calculations have also been implemented in this object.

4. Double-click **Sales Fact** to open its definition dialog.

The results appear as follows:

The screenshot shows the 'Query Subject Definition' dialog with several tabs at the top: 'Query Subject Definition', 'Filters', 'Determinants', 'Test', and 'Query Information'. The 'Query Information' tab is selected. On the left, under 'Available Model Objects', there is a tree view with 'GO Operational Model' expanded, showing 'Presentation View', 'Dimensional View', 'Consolidation View', and 'Foundation Objects View'. On the right, under 'Query Items and Calculations', there is a table with the following data:

Name	Source
ORDER_NUMBER	Original Sales & Returns Objects.ORDDEF
RETAILER_SITE_CODE	Original Sales & Returns Objects.ORDDEF
RETAILER_CONTACT_CODE	Original Sales & Returns Objects.ORDDEF
SALES_STAFF_CODE	Original Sales & Returns Objects.ORDDEF
SALES_BRANCH_CODE	Original Sales & Returns Objects.ORDDEF
ORDER_DATE	Original Sales & Returns Objects.ORDDEF
ORDER_CLOSE_DATE	Original Sales & Returns Objects.ORDDEF

Here you can see all the query items that make up the model query subject as well as any filters or determinants that might be specified. The source column tells you how the query item is derived. You can edit the definition of each query item in this dialog by clicking the ellipses in the Source column. You can also go directly to the query item definition from the Project Viewer tree. You will do this now.

5. Click **Cancel**, and then under **Sales Fact**, double-click **ORDER_NUMBER**.

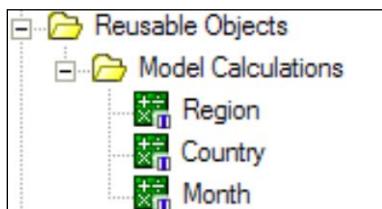
The results appear as follows:

The screenshot shows the 'Query Item Definition' dialog. It has two main sections: 'Name:' and 'Expression definition:'. The 'Name:' section contains the text 'ORDER_NUMBER'. The 'Expression definition:' section contains the text '[gosales].[ORDER_HEADER].[ORDER_NUMBER]'. There are standard Windows-style buttons for closing and applying changes at the top right.

Here you can see and edit the definition of the query item. The query item can be a direct reference to another item, or a calculation. In this case it is a direct reference that is broken down as follows: [Namespace].[Query Subject].[Query Item].

6. Click **Cancel**, and in **Foundation Objects View > gosales**, expand **Reusable Objects > Model Calculations**.

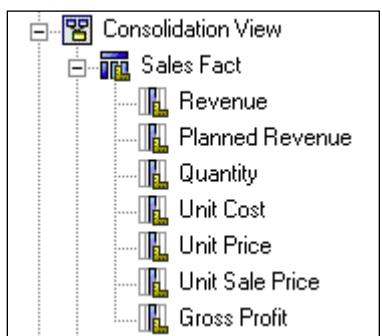
The results appear as follows:



This folder contains several commonly requested calculations (in this case, the calculation retrieves a specific language value from the data source) to allow maintenance from a single source.

7. Expand **Consolidation View > Sales Fact**.

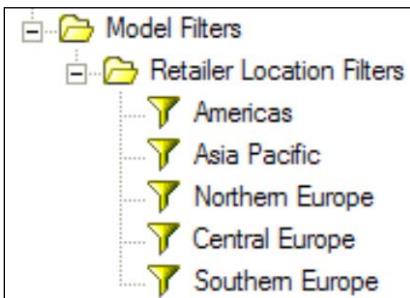
The results appear as follows:



This namespace is a simplified version of Sales Fact from the Foundation Objects View. It only contains the measures (facts), hiding the complex keys. This layer is a consistent presentation where all objects simply act as containers to consolidate information from the underlying Foundation Objects View. It also contains calculations, filters, and appropriate naming conventions. These are the objects authors will work with directly or indirectly.

8. In **Consolidation View**, expand **Model Filters > Retailer Location Filters**.

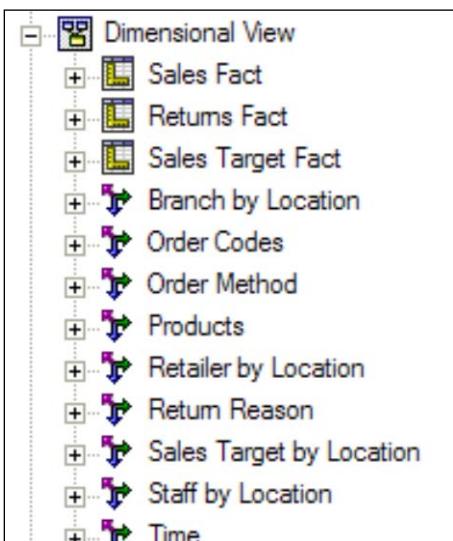
The results appear as follows:



These folders contain several filters that authors can pick from to simplify report filtering.

9. Expand **Dimensional View**.

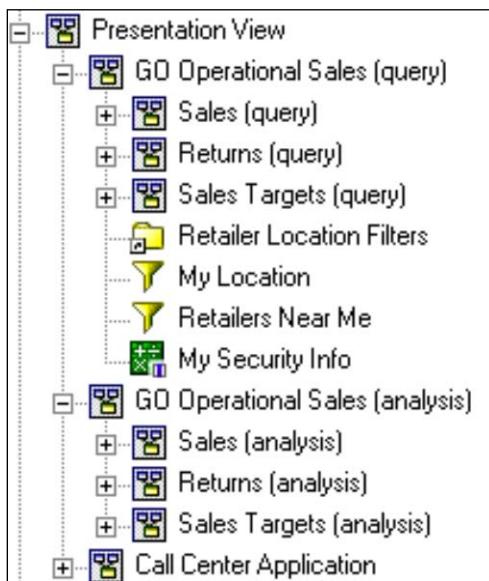
The results appear as follows:



This view comprises dimensions and measures that are based on objects in the Consolidation View. Dimensional information (hierarchies, levels, attributes, and so on). These objects are used for OLAP-style queries in Cognos Workspace, Analysis Studio, Query Studio, and Report Studio.

10. Expand **Presentation View**, and then expand **GO Operational Sales (query)** and **GO Operational Sales (analysis)**.

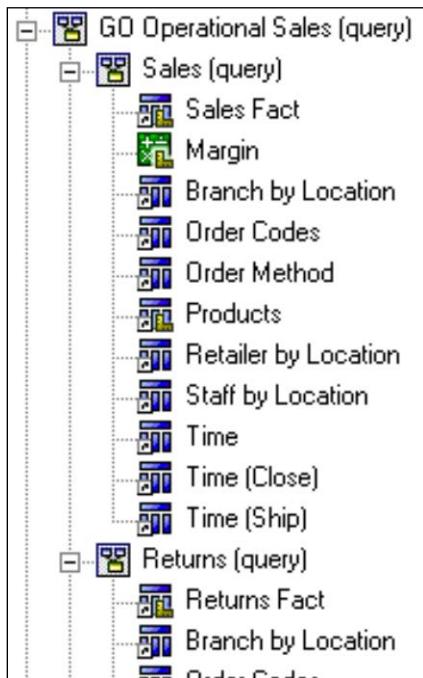
The results appear as follows:



There are two namespaces: one contains objects for queries and reports, and the other for analysis. There is also a Call Center Application namespace that is specific to a particular application to quickly retrieve order information.

11. In **GO Operational Sales (query)**, expand **Sales (query)**, and **Returns (query)**.

The results appear as follows:



This material is meant for IBM Academic Initiative use only. NOT FOR RESALE

These two namespaces contain shortcuts that logically group the Consolidation View objects. Facts are grouped with their related dimensions. The namespaces have some overlapping object names (conformed dimensions). To create multi-fact queries across multiple areas of the business, include at least one conformed dimension.

Task 3. View the package as a report author.

1. Expand **Packages**, and then double-click **GO Operational (query)**.

The results appear as follows:



The package contains a subset of the project specific to objects to be used for relational queries and reports.

2. Click **Cancel**, right-click **GO Operational (query)**, and then click **Publish Packages**.
3. Click **Next** twice, clear **Verify the package before publishing**, and then click **Publish**.
4. Click **Finish**.
5. Open **Internet Explorer** and click the **IBM Cognos 10** link.
This opens the following page: <http://localhost:88/ibmcognos>.
6. Log on as **admin/Education1**.
This will log you in to IBM Cognos Connection.
7. Click **Author business reports**, and then click **GO Operational (query)**.
This opens the package you just published in IBM Cognos Workspace Advanced.
8. Click **Create New**, select **List** and then click **OK**.

9. In the **Source** pane, expand **Sales (query)** and **Returns (query)**. Notice that this is the same structure and grouping seen in the Presentation View in the previous task.
10. Add the following items to the report:

Query Subject	Query Item
Sales (query) > Products	Product Line
Sales (query) > Time	Year
Sales (query) > Sales Fact	Revenue
Returns (query) > Returns Fact	Return Quantity

Tip: To add a query item to the report, you can either double-click the query item in the Source pane, or click and drag the item onto the report area.

11. Click the **Product Line** column heading, and then click **Group/Ungroup**  to group the report on **Product Line**.

The results appear as follows:

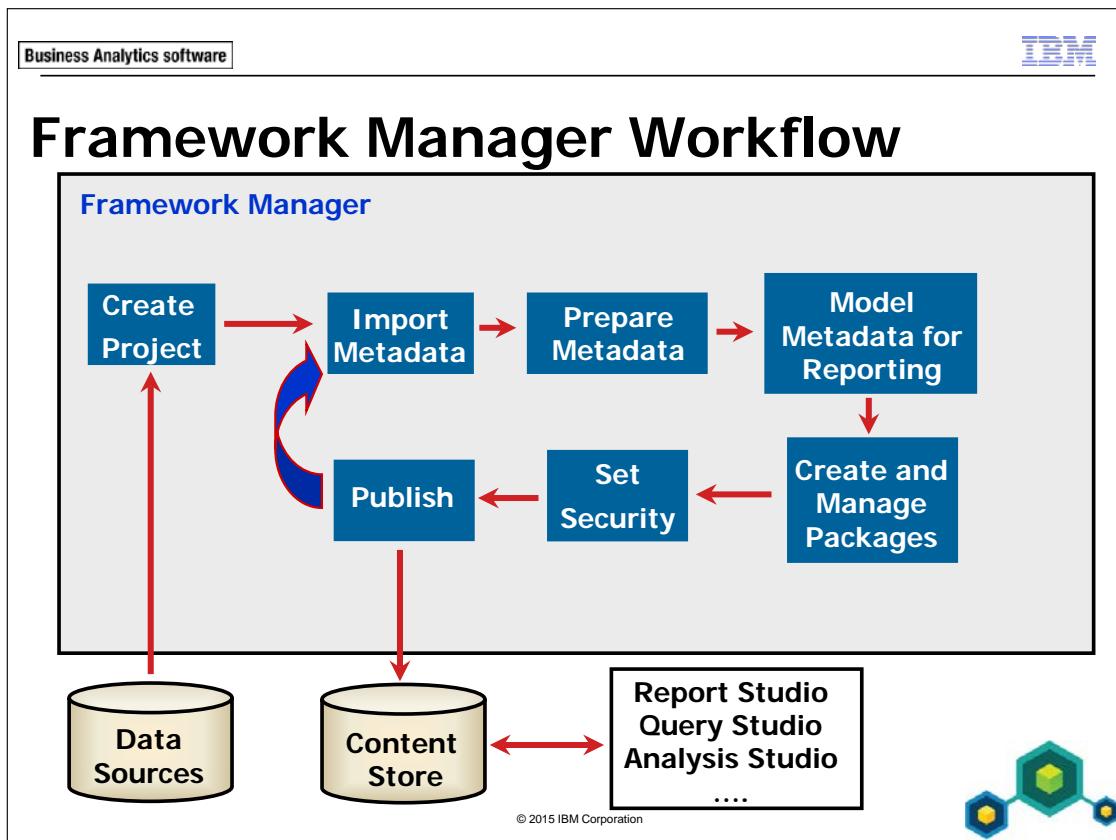
Product Line	Year	Revenue	Return Quantity
Camping Equipment	2010	\$332,986,338.06	60,966
	2011	\$402,757,573.17	66,256
	2012	\$500,382,422.83	97,617
	2013	\$352,910,329.97	79,604
Golf Equipment	2010	\$153,553,850.98	8,850
	2011	\$168,006,427.07	14,599
	2012	\$230,110,270.55	21,238
	2013	\$174,740,819.29	8,560
Mountaineering Equipment	2011	\$107,099,659.94	25,808
	2012	\$161,039,823.26	53,137
	2013	\$141,520,649.70	27,188
Outdoor Protection	2010	\$36,165,521.07	222,611
	2011	\$25,008,574.08	68,957
	2012	\$10,349,175.84	31,225

This material is meant for IBM Academic Initiative use only. NOT FOR RESALE

12. Close the browser without saving the report.
13. In **Framework Manager**, from the **File** menu, click **Close** to close the project without saving changes.

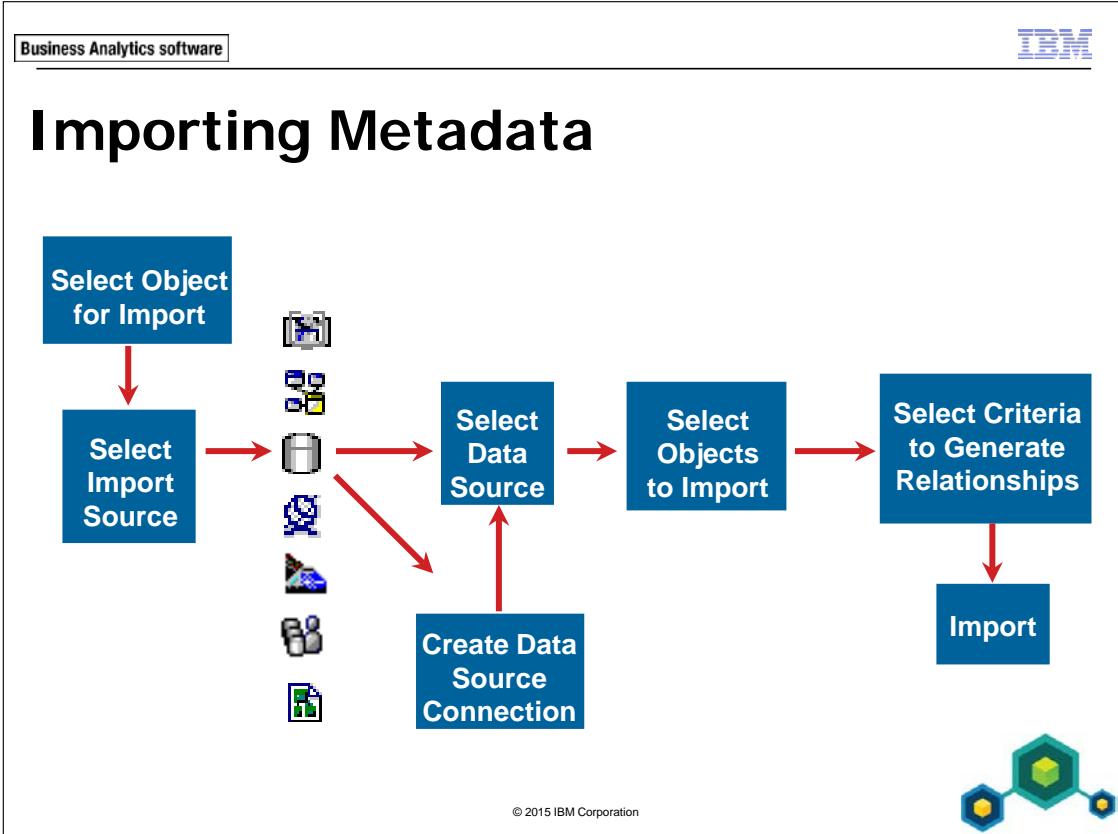
Results:

You explored the Framework Manager UI and saw some of the final results in the model that you will build throughout this course. You also saw the benefits of creating a simple-to-use view of the metadata for the business.



- **Import Metadata** - import objects such as tables, views, and procedures
- **Prepare Metadata** - examine and modify properties and relationships
- **Model Metadata for Reporting** - add business value specific to reporting requirements and model for predictable results
- **Create and Manage Packages** - identify subsets of the metadata to be published
- **Set Security** - apply security at various levels to restrict access
- **Publish** - publish packages to the IBM Cognos servers for use by report authors and business analysts

The blue arrow in the above diagram emphasizes that modeling is a reiterative process. Managing the project is another important activity, not shown on the diagram because it is to be performed throughout the modeling process. This includes activities such as implementing multi-user modeling, sharing and reusing information, action logging and synchronizing, and validating the project.



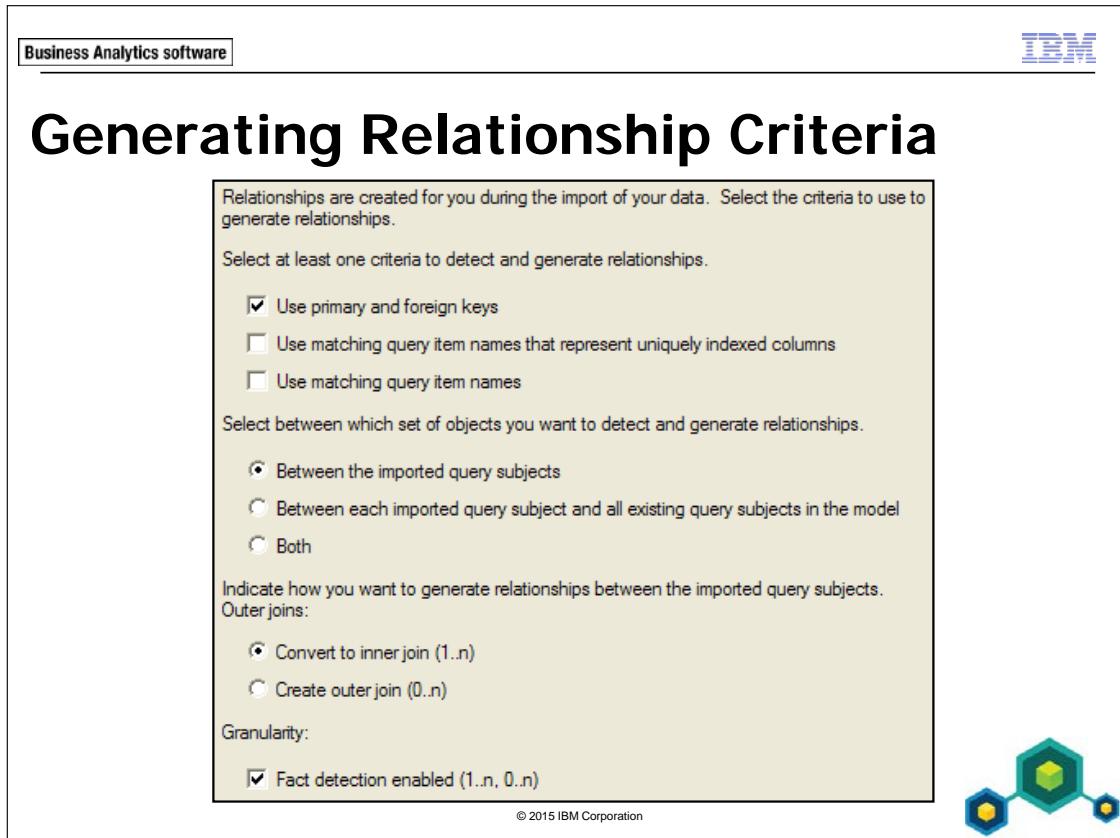
Import only the required metadata in a phased approach to minimize clutter in the model.

For relational data sources, choose the model object into which you will import, and then do the following:

1. Select Data Sources as the import source.
2. Select the data source to be used (usually a relational database), or create a data source connection, and then select it.

Note: Creating a data source connection is usually done by an Administrator, using IBM Cognos Administration. Depending on the roles and responsibilities in your organization, you may need to request that your Administrator perform this task for you.

3. Select the required objects to import; for example, tables, views, or procedures.
4. Select the criteria by which Framework Manager will generate relationships.



This dialog box is part of the Import Metadata Wizard.

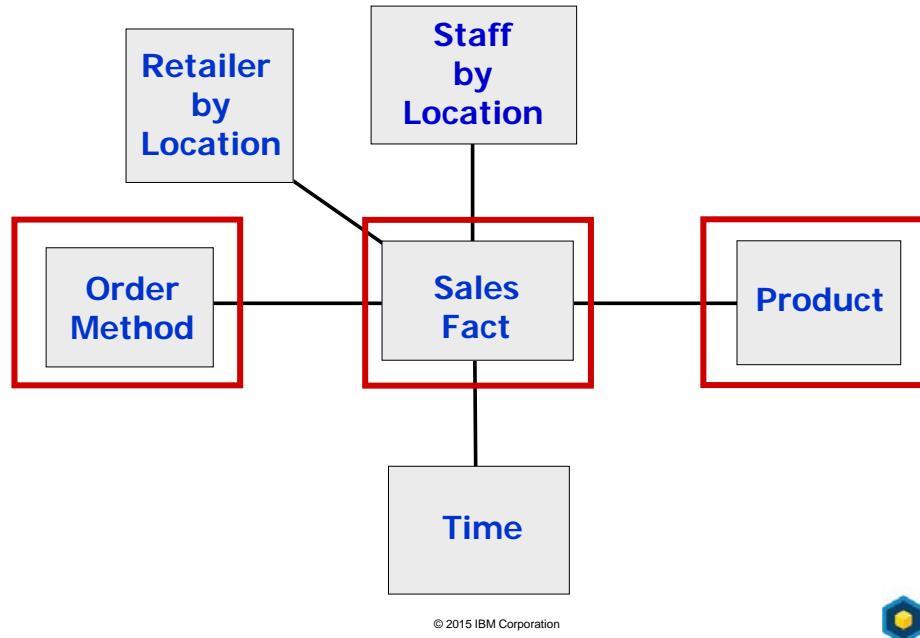
In general, use the first check box for relational data from the same database. The other two check boxes are often used when importing from a different database.

You can request to create relationships among either the objects being imported, or the objects being imported and the existing objects in the model, or both.

By default, an import converts outer joins to inner joins. This is done for performance reasons. You can choose to generate outer joins if that meets your business needs or you can edit specific relationships after import to meet your needs.

Another option is to enable or disable fact detection. If this option is disabled, all relationships will be 1..1 to 1..1.

Requirements Review



In the next demo, you will import metadata to satisfy the highlighted reporting requirements above. The database tables used to satisfy each requirement are as follows:

Reporting Requirement	Database Tables
Sales Fact	ORDER_DETAILS ORDER_HEADER
Order Method	ORDER_METHOD
Product	PRODUCT PRODUCT_LINE PRODUCT_NAME_LOOKUP PRODUCT_TYPE

Demo 2: Create a Baseline Project

Purpose:

Your first step as a data modeler for the Sample Outdoors Company is to create a new project, set up a data source connection, and start the metadata modeling process.

Component: **Framework Manager**

Project: **GO Operational**

Task 1. Create a project.

1. If necessary, launch **Framework Manager**, and under **Projects**, click **Create a new project**.

2. In the **Location** box, click **Browse**, select **C:\Edcognos\B5A52\Course_Project**, click **OK**, and then in the **Project name** box, type **GO Operational**.

The GO Operational folder is created by default and appears in the Location box.

3. Deselect the **Use Dynamic Query Mode** check box, and then click **OK**.

4. If prompted, log in as User ID **admin**, and Password **Education1**.

The Select Languages dialog box appears. You will initially set the default and design language for this project as English.

5. Ensure that **English** is selected, and then click **OK**.

The Metadata Wizard appears. You will first organize your project and then import metadata into the appropriate location in a later task.

6. Click **Cancel**.

The project opens in Framework Manager. Notice in the Project Viewer that the project appears with the name you provided and a default namespace called Model.

Task 2. Create and organize objects.

1. Rename the **Model**  namespace to **GO Operational Model**.

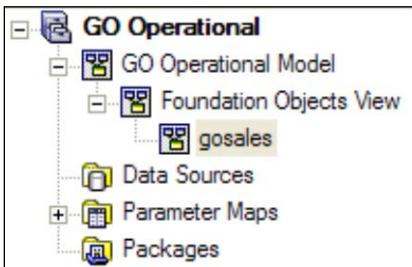
Tip: You can press F2 to edit the label.

2. Right-click the **GO Operational Model** namespace, point to **Create**, and then click **Namespace**.

3. Rename **New Namespace** to **Foundation Objects View**.

4. Right-click the **Foundation Objects View** namespace, point to **Create**, and then click **Namespace**.
5. Rename **New Namespace** to **gosales**.

The model appears as shown below:



Creating the **gosales** namespace makes the Foundation Objects View extensible in the event you want to import metadata from another data source. At the time of the import of the second data source, you would create another namespace to contain those data source query subjects.

You will now begin the process for importing the metadata into the **gosales** namespace. You will start by creating a data source and testing the connection to it.

Task 3. Create a data source and test the connection.

1. Right-click the **gosales** namespace, and then click **Run Metadata Wizard**.
The Metadata Wizard appears. From here you can select the source from which you will import the metadata. You will import data from a relational database.
2. Ensure that **Data Sources** is selected, and then click **Next**.
You are now prompted to select a data source from a list of available data sources. You can create data source connections in either IBM Cognos Administration, or in Framework Manager.
In order to demonstrate this process, you will create a new data source named **GOSALES2**. The majority of the course uses a pre-defined data source connection (**GOSALES**), which connects to the same underlying database as **GOSALES2**.
3. Click **New**.
The Welcome page of the New Data Source Wizard appears. This is the same wizard that is used to create data sources through the Directory tool in the IBM Cognos Connection administration interface.
4. Click **Next**, in the **Name** box, type **GOSALES2**, and then click **Next**.

5. In the **Type** list, click **IBM DB2**, clear the **Configure JDBC connection** check box, and then click **Next**.

When you select the Configure JDBC connection check box, you are prompted to enter additional connection details that support Dynamic Query Mode. This feature is discussed in detail in Optimize and Tune Framework Manager Models.

6. In the **DB2 database name** box, type **GS_DB**.
7. Under **Signons**, select the **Password** check box, in the **User ID** box, type **GOSALES**, and then in the **Password** and **Confirm password** boxes, type **Education1**.
8. Click **Test the connection**, and then click **Test**.

The View the results - Test the connection page appears indicating that the test succeeded.

9. Click **Close**, and then click **Close** again.
10. Click **Finish**, and then click **Close**.

A new data source called GOSALES2, through which you will import metadata, now appears in the list.

You will now take a moment to view this data source in IBM Cognos Connection.

11. Launch **IBM Cognos Connection**, log on, click **Administer IBM Cognos content**, and then click the **Configuration** tab.
- The GOSALES2 data source appears in the list. You could have also created the data source here rather than through Framework Manager. If you need to edit the data source connection, you will need to do it in this location.
12. Close **IBM Cognos Connection**.

Task 4. Import metadata.

1. In **Framework Manager**, ensure the newly created **GOSALES2** data source is selected, and then click **Next**.
2. Under **GOSALES2**, expand **GOSALES > Tables**.

3. Select the following initial tables:

ORDER_DETAILS
ORDER_HEADER
ORDER_METHOD
PRODUCT
PRODUCT_LINE
PRODUCT_NAME_LOOKUP
PRODUCT_TYPE

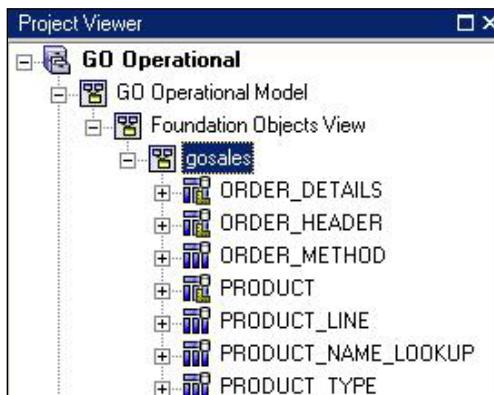
4. Click **Next**, leave the defaults for the **Generate Relationship** criteria, and then click **Import**.

The import process begins, and then a message appears summarizing the seven query subjects and six relationships that were imported.

5. Click **Finish**.

6. In the **Project Viewer** pane, expand the **gosales** namespace.

The results appear as follows:



The namespace now contains a list of data source query subjects, which represent each of the tables that were imported from the relational database.

7. Expand the **Data Sources** folder.

This folder now contains the GOSALES2 data source, which you specified during the import. This data source is now associated with the objects you just imported.

8. Select the **GOSALES2** data source.

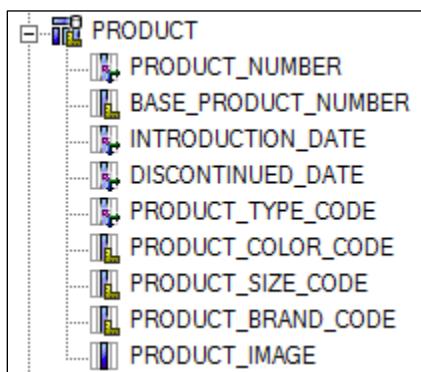
9. In the **Properties** pane, ensure the **Query Processing** property is set to **Limited Local**.

This will allow IBM Cognos to process operations locally that are not supported by the data source.

Task 5. Test basic data access.

1. In the **Project Viewer**, expand **PRODUCT**.

The results appear as follows:



Here you see all the query items for this query subject. These query items represent the columns found in the data source table.

2. Double-click **PRODUCT**.

The SQL tab shows "Select * from [GOSALES2].PRODUCT". This SQL statement defines the scope of the PRODUCT query subject. It is used to generate your query item list as well as the run time SQL when you create a query using the query items. The [GOSALES2] portion is the name of the data source connection with which the query subject is associated.

3. Click the **Test** tab, and then click **Test Sample** in the bottom-right corner.

The results appear as follows:

Query Subject Definition - PRODUCT

Test results						
PRODUCT_NUMBER	BASE_PRODUCT_NUMBER	INTRODUCTION_DATE	DISCONTINUED_DATE	PRODUCT_TYPE_CODE	PRODUCT_COLOR_CODE	P
1110	1	Feb 15, 2001 12:00:00 AM		951	908	80
2110	2	Feb 15, 2001 12:00:00 AM		951	906	80
3110	3	Feb 15, 2001 12:00:00 AM		951	924	82
4110	4	Feb 15, 2001 12:00:00 AM		951	923	80
5110	5	Feb 15, 2001 12:00:00 AM		951	923	82
6110	6	Mar 5, 2003 12:00:00 AM		951	923	82
7110	7	Feb 15, 2001 12:00:00 AM		951	923	84
8110	8	Mar 5, 2003 12:00:00 AM		951	912	84
9110	9	Mar 5, 2003 12:00:00 AM		951	900	80

Use Dynamic Query Mode

The test sample contains the first 25 records, confirming your access to the data. You can increase or decrease the amount of rows retrieved in the Options located in the lower right corner. If you want to know how many rows of data there are in the data source table, click Total Rows.

This material is meant for IBM Academic Initiative use only. NOT FOR RESALE

- Click the **Query Information** tab.

The results appear as follows:

Cognos SQL
<pre> select PRODUCT.PRODUCT_NUMBER as PRODUCT_NUMBER, PRODUCT.BASE_PRODUCT_NUMBER as BASE_PRODUCT_NUMBER, PRODUCT.INTRODUCTION_DATE as INTRODUCTION_DATE, PRODUCT.DISCONTINUED_DATE as DISCONTINUED_DATE, PRODUCT.PRODUCT_TYPE_CODE as PRODUCT_TYPE_CODE, PRODUCT.PRODUCT_COLOR_CODE as PRODUCT_COLOR_CODE, PRODUCT.PRODUCT_SIZE_CODE as PRODUCT_SIZE_CODE, PRODUCT.PRODUCT_BRAND_CODE as PRODUCT_BRAND_CODE, PRODUCT.PRODUCT_IMAGE as PRODUCT_IMAGE from GOSALES2..GOSALES.PRODUCT PRODUCT </pre>
Native SQL
<pre> select "PRODUCT"."PRODUCT_NUMBER" "PRODUCT_NUMBER" , "PRODUCT"."BASE_PRODUCT_NUMBER" "BASE_PRODUCT_NUMBER" , "PRODUCT"."INTRODUCTION_DATE" "INTRODUCTION_DATE" , "PRODUCT"."DISCONTINUED_DATE" "DISCONTINUED_DATE" , "PRODUCT"."PRODUCT_TYPE_CODE" "PRODUCT_TYPE_CODE" , "PRODUCT"."PRODUCT_COLOR_CODE" "PRODUCT_COLOR_CODE" , "PRODUCT"."PRODUCT_SIZE_CODE" "PRODUCT_SIZE_CODE" , "PRODUCT"."PRODUCT_BRAND_CODE" "PRODUCT_BRAND_CODE" , "PRODUCT"."PRODUCT_IMAGE" "PRODUCT_IMAGE" from "GOSALES"."PRODUCT" "PRODUCT" FOR FETCH ONLY </pre>

This tab displays the full SQL in two formats: the generic Cognos syntax (Cognos SQL) and the syntax specific to the data source (Native SQL). This is the SQL generated at run time and passed to the data source. The SQL generated here is based on the SQL defined on the SQL tab of the data source query subject.

- Click **OK**.

Task 6. Change SQL generation settings and specify a function list.

- From the **Project** menu, click **Edit Governors**.

These properties govern the queries that are generated based on the model. These will be described in more detail later in the course.

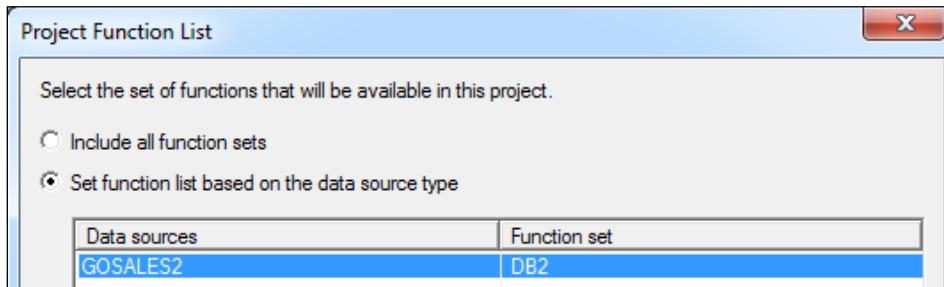
- Clear the **Use WITH clause when generating SQL** check box, and then click **OK**.

You will now specify a function list specific to the data source you are using in this model. These functions are specific to the database vendor and allow authors and modelers to perform advanced tasks with the data sources such as date and string manipulations.

Specifying a function set per data source will limit the function sets that are published to IBM Cognos with a package. By default, all supported function sets are published with a package. Publishing only the required functions sets can reduce the package size and prevent confusion where authors may inadvertently use functions from another data source.

3. From the **Project** menu, click **Project Function List**.
4. Select **Set function list based on the data source type**, and then in the **Function set** column beside **GOSALES2**, select **DB2**.

The results appear as follows:



Now, by default, only DB2 functions will be published with a package in this project.

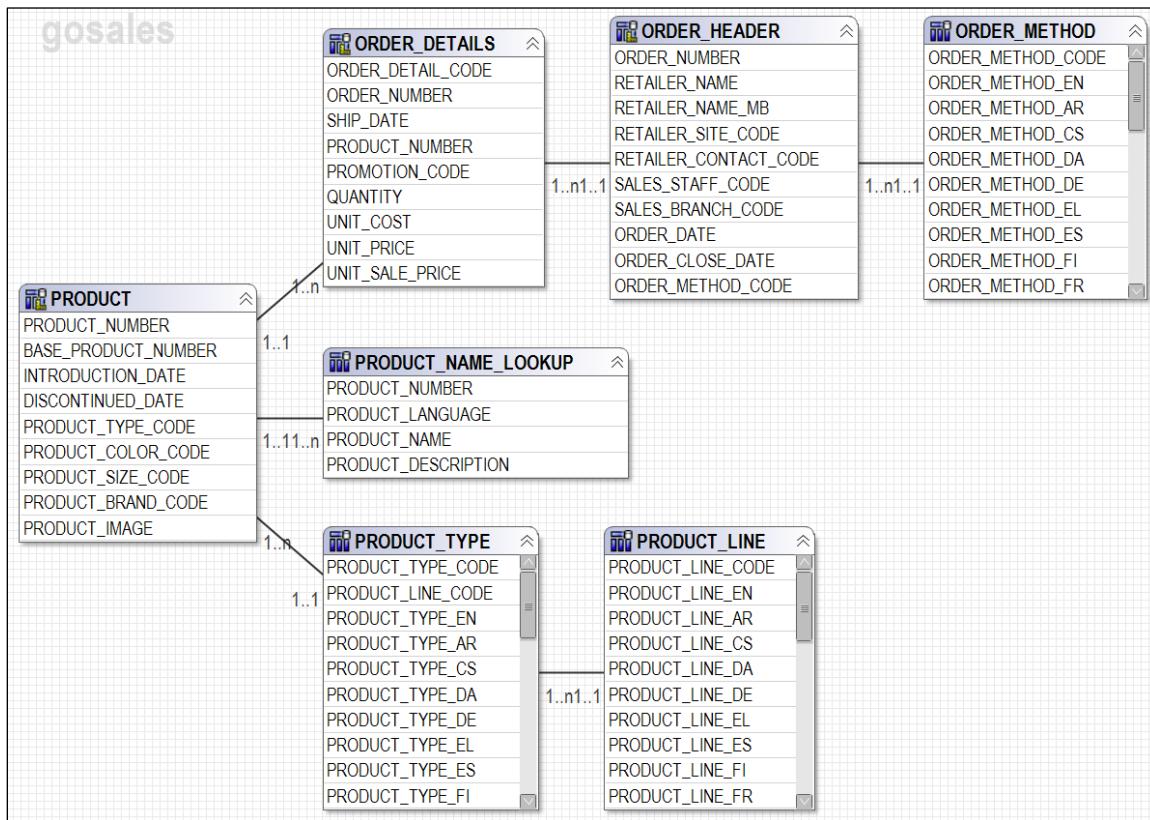
5. Click **OK**.

Task 7. Test multiple query subject data access.

1. If necessary, in the center pane click **Diagram**.
2. Close the **Properties** and **Tools** panes to view more of the center pane diagram, and then double-click the **gosales** box to focus on the **gosales** namespace.

3. Click **Fit All** .

The results appear as follows:



This pane displays all of the relationships generated by Framework Manager during import. These relationships allow IBM Cognos to generate the appropriate SQL at run time in order to join query items from the different query subjects. You can adjust the settings of the diagram from the Diagram menu. For example, you can choose to view cardinalities in Merise notation (as seen in the screen capture) or Crowsfeet notation (provides a pictorial representation of the relationship). Take a moment to explore the options.

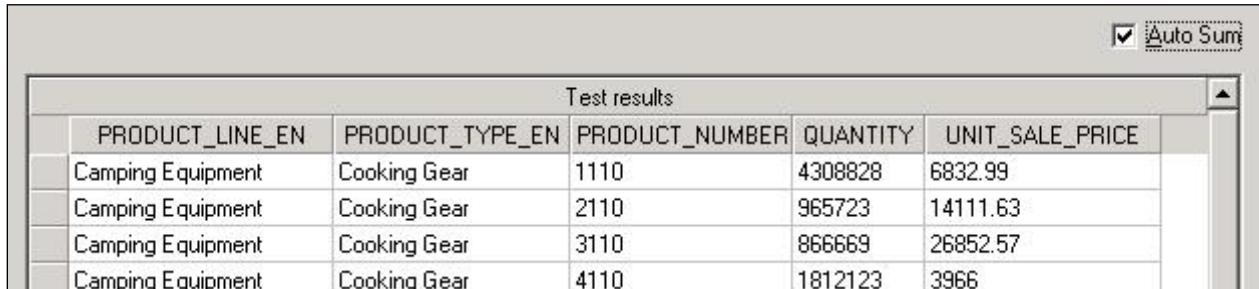
4. In the **diagram**, Ctrl-click to select the following query items:

Query Subject	Query Item
PRODUCT_LINE	PRODUCT_LINE_EN
PRODUCT_TYPE	PRODUCT_TYPE_EN
PRODUCT	PRODUCT_NUMBER
ORDER_DETAILS	QUANTITY UNIT_SALE_PRICE

This material is meant for IBM Academic Initiative use only. NOT FOR RESALE

5. Right-click one of the selected items, click **Test**, and then click **Test Sample**. A sample 25-row report confirms that you can access the data. Note the values for quantity and unit sale price.
6. In the top right corner of the window, click **Auto Sum**, and then click **Test Sample** again.

The results appear as follows:



The screenshot shows a 'Test results' dialog box with a table containing four rows of data. The table has columns labeled PRODUCT_LINE_EN, PRODUCT_TYPE_EN, PRODUCT_NUMBER, QUANTITY, and UNIT_SALE_PRICE. The data is as follows:

Test results				
PRODUCT_LINE_EN	PRODUCT_TYPE_EN	PRODUCT_NUMBER	QUANTITY	UNIT_SALE_PRICE
Camping Equipment	Cooking Gear	1110	4308828	6832.99
Camping Equipment	Cooking Gear	2110	965723	14111.63
Camping Equipment	Cooking Gear	3110	866669	26852.57
Camping Equipment	Cooking Gear	4110	1812123	3966

This time, you see that QUANTITY has been summed. UNIT_SALE_PRICE has also been summed, which we will correct later.

7. Close the **Test Results** dialog box, and then save the project.

Results:

You started to develop the model that will support the business requirements of report and ad hoc query authors. You created a project, structured it, and imported baseline metadata.

Demo 3: Publish a Package

Purpose:

Now that you have a baseline project, you will verify its reporting capabilities by publishing a package and using it to view data in IBM Cognos Workspace Advanced.

Components: **Framework Manager, IBM Cognos Workspace Advanced**

Project: **GO Operational**

Package: **GO Operational**

Task 1. Create and publish a package.

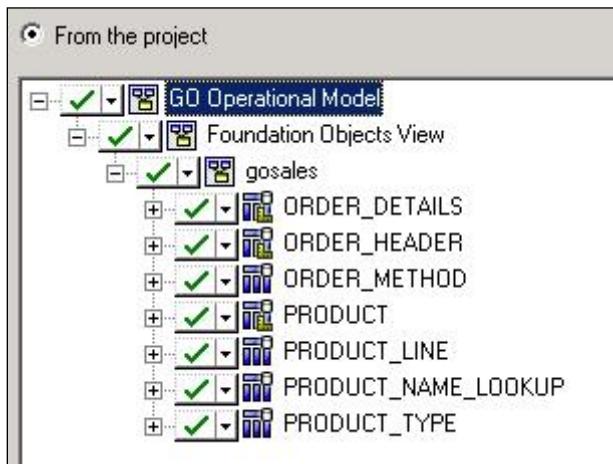
1. In the **Project Viewer** pane, right-click **Packages**, point to **Create**, and then click **Package**.

The Create Package Wizard appears. Since the package will include the full contents of the project, you will give it the same name as the project.

Note: To be able to publish packages, a user must belong to the Report Administrators role, or have write privileges to the Public Folders area of Cognos Connection.

2. In the **Name** box, type **GO Operational**, and then click **Next**.
3. Expand **Foundation Objects View** and **gosales**.

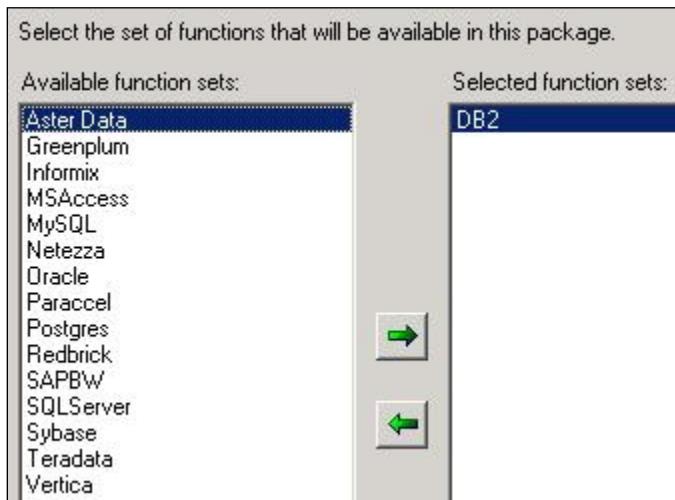
The results appear as follows:



This is where you can restrict the package contents to a subset of your model.

4. Click **Next.**

The results appear as follows:



Here you have the option to specify which function sets get published with the package. Since you associated DB2 functions with the GOSALES data source, it is by default, the only function set currently selected.

5. Click **Finish.**

A dialog box appears indicating that the package was created successfully, and prompts you to open the Publish Package wizard. Publishing places a package in the IBM Cognos Content Store, where it can be accessed from IBM Cognos Connection and the IBM Cognos Studios.

6. Click **Yes.**

IBM Cognos allows you to maintain multiple versions of a package. This is done to prevent breaking reports when re-publishing a package with changes that affect the reports. You will not use this feature for most of this course.

7. Clear the **Enable model versioning check box, accept the remaining defaults, and then click **Next**.**

8. Click **Next, and then click **Publish**.**

The package is verified by default before the package is published.

Once verified and published, a message appears within the wizard indicating that the package was successfully published.

9. Click **Finish to close the wizard, and then save the project.**

Task 2. View the package in IBM Cognos Workspace Advanced.

1. Open **Internet Explorer**, click the **IBM Cognos 10** link.

This opens the log on page at <http://localhost:88/ibmcognos>.

2. Log on as **admin/Education1**.

The Welcome page of IBM Cognos Connection appears.

3. Under **My Actions**, click **Author business reports**.

The Select a package page appears, and prompts you to choose a package to use in creating the report. Note that the GO Operational model you just published is available.

4. Click **GO Operational**.

5. In the **Welcome** dialog, click **Create New**.

6. In the **New** dialog, click **List** and then click **OK**.

IBM Cognos Workspace Advanced appears in the browser. In the Source pane, you can see that the GO Operational root namespace and the Foundation Objects View namespace appear. These were the objects you specified in the Publish Package wizard in Framework Manager.

7. Expand **Foundation Objects View**, and then **gosales**.

The query subjects appear and are available to an ad hoc query author for creating a report.

Task 3. Create a report.

1. In the **gosales** namespace, expand the **PRODUCT_LINE** query subject, and then double-click the **PRODUCT_LINE_EN** query item.
2. Repeat step 1 to add the following items:

Query Subject	Query Item
PRODUCT_TYPE	PRODUCT_TYPE_EN
PRODUCT	PRODUCT_NUMBER
ORDER_DETAILS	QUANTITY UNIT_SALE_PRICE

The results appear as follows:

PRODUCT_LINE_EN	PRODUCT_TYPE_EN	PRODUCT_NUMBER	QUANTITY	UNIT_SALE_PRICE
Camping Equipment	Cooking Gear	1110	4,308,828	6,832.99
Camping Equipment	Cooking Gear	2110	965,723	14,111.63
Camping Equipment	Cooking Gear	3110	866,669	26,852.57
Camping Equipment	Cooking Gear	4110	1,812,123	3,966
Camping Equipment	Cooking Gear	5110	813,780	62,344.6
Camping Equipment	Cooking Gear	6110	442,136	144,169.88
Camping Equipment	Cooking Gear	7110	686,493	74,157.07
Camping Equipment	Cooking Gear	8110	245,559	190,834.99

You have successfully created the same ad hoc query you tested in Framework Manager earlier. Here the values are automatically sorted and summarized.

3. Close **Internet Explorer**, do not save changes, and leave **Framework Manager** open for the next demo.

Results:

You have verified the reporting capabilities of your new project by publishing a package from it and using the package to view data in IBM Cognos Workspace Advanced.

Importing Additional Metadata

Relationships are created for you during the import of your data. Select the criteria to use to generate relationships.

Select at least one criteria to detect and generate relationships.

- Use primary and foreign keys
- Use matching query item names that represent uniquely indexed columns
- Use matching query item names

Select between which set of objects you want to detect and generate relationships.

- Between the imported query subjects
- Between each imported query subject and all existing query subjects in the model
- Both

Indicate how you want to generate relationships between the imported query subjects.
Outer joins:

- Convert to inner join (1..n)
- Create outer join (0..n)

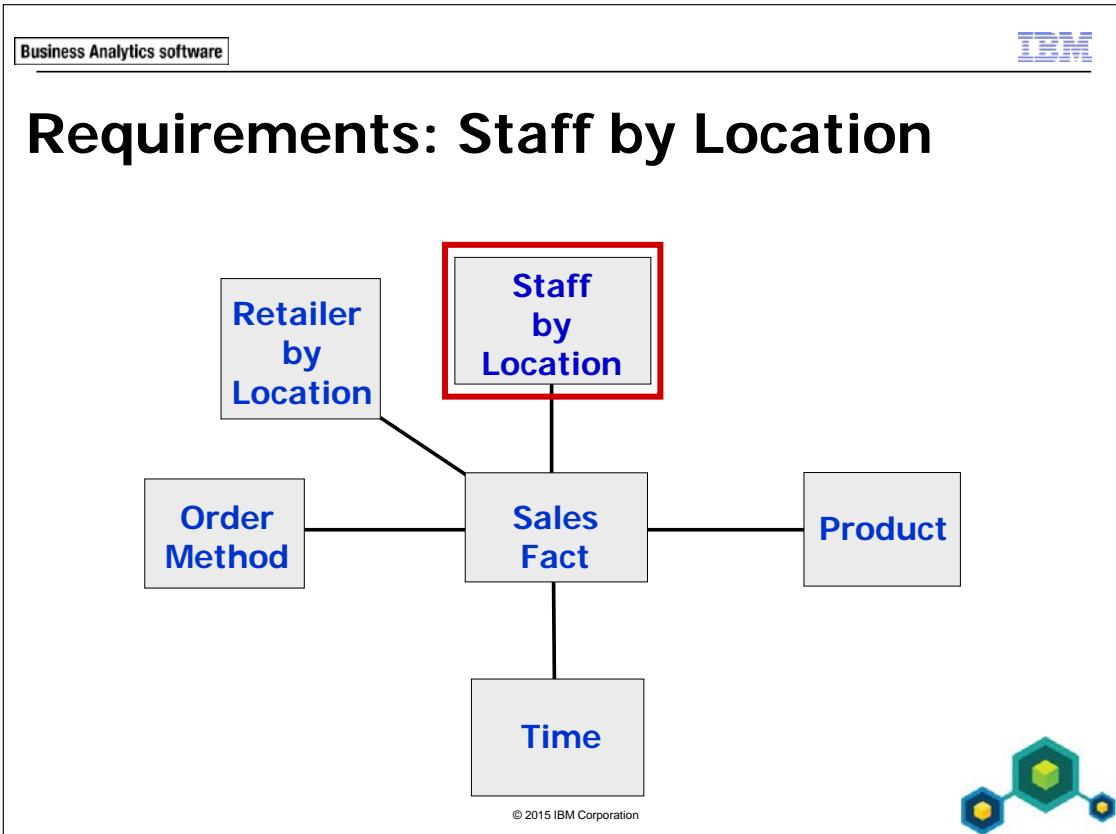
Granularity:

- Fact detection enabled (1..n, 0..n)

© 2015 IBM Corporation



When importing additional metadata from the same data source or another compatible data source (primary and foreign keys match), select Both in the wizard. This generates relationships among the objects being imported as well as the existing objects in the model. By selecting this option, you will not need to create the relationships manually after import.



In the next demo, you will import metadata to satisfy the highlighted reporting requirement above. The database tables used to satisfy this requirement are as follows:

Reporting Requirement	Database Tables
Staff by Location	BRANCH COUNTRY SALES_REGION EMPLOYEE EMPLOYEE_HISTORY POSITION_LOOKUP

This material is meant for IBM Academic Initiative use only. NOT FOR RESALE

Demo 4: Extend the Model to Add Staff Location Metadata

Purpose:

Now that you have an understanding of the Framework Manager modeling environment and process, you can extend your model to support additional business requirements from report and ad hoc query authors, namely reporting sales by sales staff location.

Component: **Framework Manager**

Project: **GO Operational**

Task 1. Import additional metadata.

1. In the **Project Viewer**, expand **GO Operational Model > Foundation Object view**.
2. Right-click the **gosales** namespace, and then click **Run Metadata Wizard**. You will import more items from the GOSALES database.
3. Ensure that **Data Sources** is selected, and then click **Next**.
4. Ensure that **GOSALES** is selected, and then click **Next**.
5. In the list of objects, expand **GOSALES > Tables**, and then select the following tables:

BRANCH

COUNTRY

SALES_REGION

6. Expand **GOSALESHR** and **Tables**, and then select the following tables:

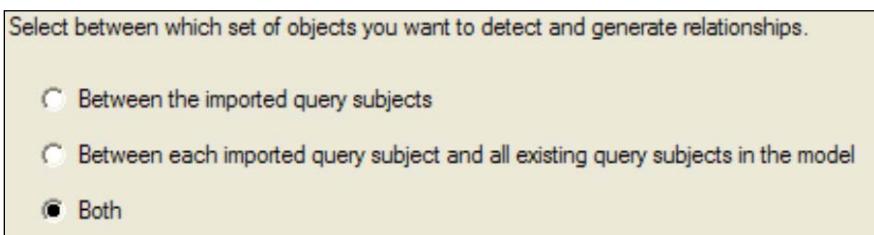
EMPLOYEE

EMPLOYEE_HISTORY

POSITION_LOOKUP

7. Click **Next**.

8. On the **Generate Relationships** page, click the **Both** radio button as shown below.



You want to generate relationships among the new query subjects as well as between the new and existing query subjects. This will prevent you from having to manually create the relationships after the import.

9. Click **Import**.

The import process begins, and then a message appears summarizing the count of objects that were imported.

10. Click **Finish**.

The gosales namespace now contains a list of query subjects, which represent each of the tables that were imported from the relational database.

11. Expand the **Data Sources** folder, and then click **GOSALES1**.

The second data source, GOSALES1, was created for the query subjects you selected from the second schema, GOSALESHR.

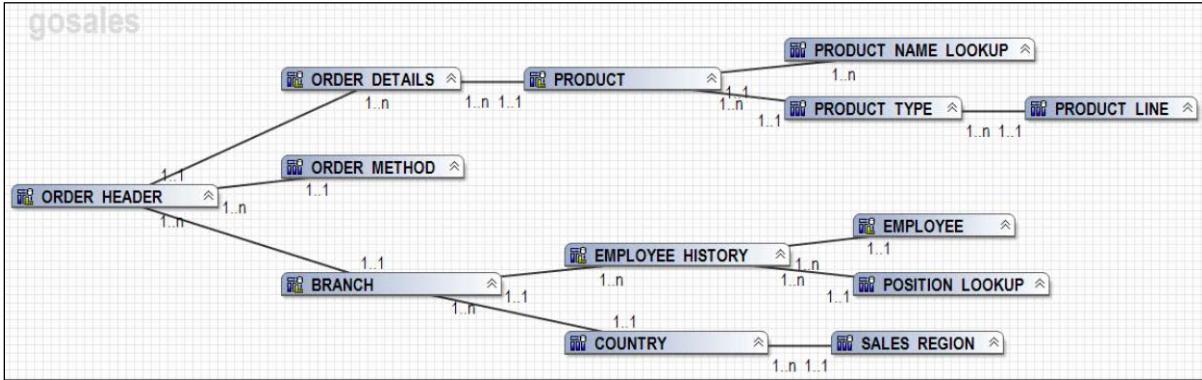
12. In the **Properties** pane, note that the **Schema** property is set to **GOSALESHR**.

Tip: To open the **Properties** pane, from the **View** menu, click **Properties**.

Task 2. Rearrange the diagram.

- From the **Diagram** menu, click **Diagram settings**.
- Under **Level of Details**, deselect **Query Items**, and then click **OK**.
- Close the **Properties** pane, ensure **Diagram** is selected and then, from the toolbar, click **Auto Layout**

4. Beside Layout Style, ensure that **Standard** is selected, and then set both **Horizontal** and **Vertical** distances to **30**, click **Apply**, and then click **Close**. The results appear as follows:



You can now view all the items imported into your project in a simplified way by reducing the detail and adjusting the space between the objects.

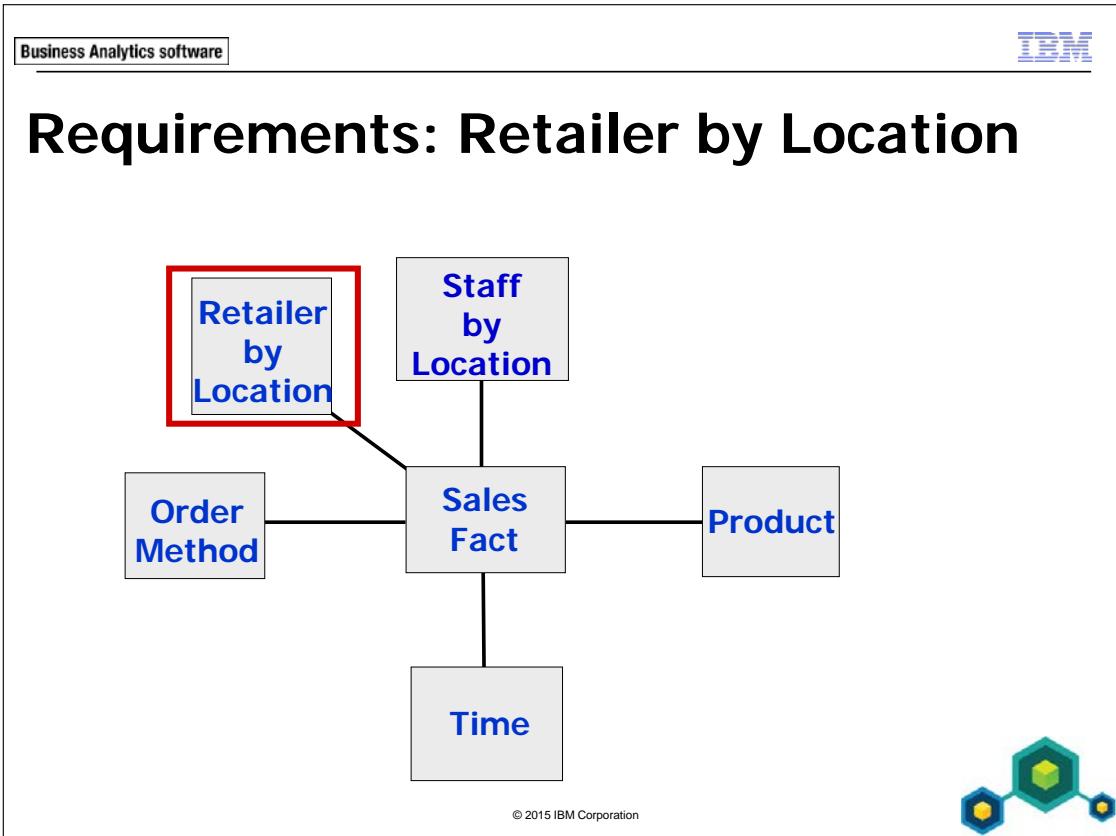
If you were importing a star schema structure, you would select the Star option under Layout Style to set fact query subjects as focal points with their related dimensions surrounding them.

You will identify and correct any missing relationships in the next module.

5. Leave Framework Manager open for the Workshop.

Results:

You extended the model to support additional business requirements for report and ad hoc query authors by importing metadata to support sales staff location queries.



In the next workshop, you will import metadata to satisfy the highlighted reporting requirement above. The database tables used to satisfy this requirement are as follows:

Reporting Requirement	Database Tables
Retailer	RETAILER RETAILER_SITE RETAILER_TYPE

Summary

- You should now be able to:
 - follow the IBM Cognos and Framework Manager workflow processes
 - define a project and its structure
 - describe the Framework Manager environment
 - create a baseline project
 - enhance the model with additional metadata

© 2015 IBM Corporation

Workshop 1: Enhance the Model to Add Retailer Metadata

Business analysts at the Sample Outdoors Company have additional reporting requirements that need to be met. They want to report on sales by retailer.

To support this:

- Import the following tables into the model, and rearrange the diagram to show all query subjects:
 - RETAILER
 - RETAILER_SITE
 - RETAILER_TYPE
- Use the Test tool in Framework Manager to verify that you can report on the new data.

For more information about where to work and the workshop results, refer to the Tasks and Results section that follows. If you need more information to complete a task, refer to earlier demos for detailed steps.

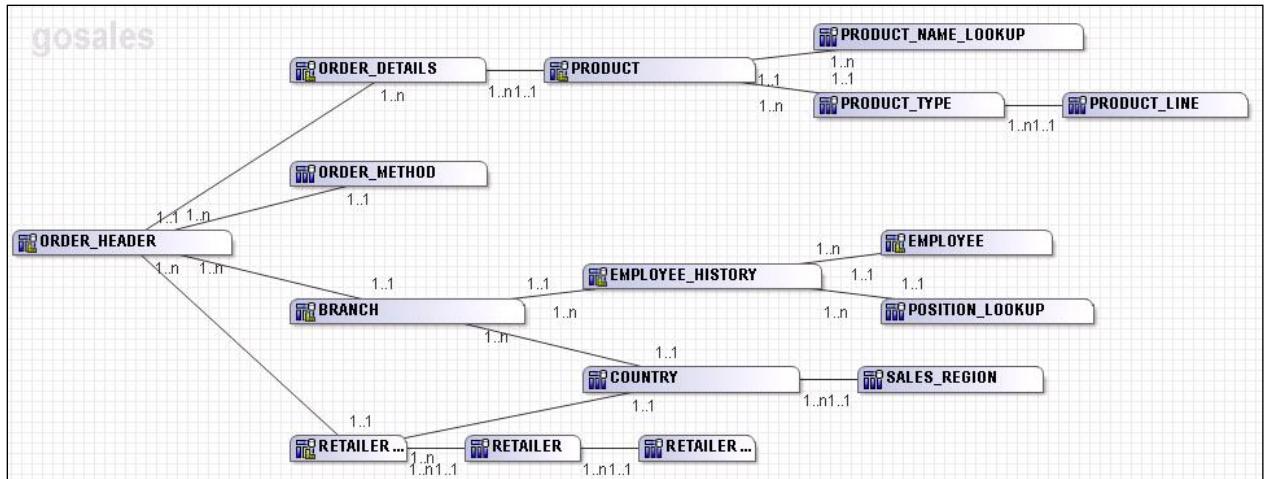
Workshop 1: Tasks and Results

Task 1. Import additional metadata.

- In **GO Operational Model > Foundation Objects View**, right-click the **gosales** namespace, and click **Run Metadata Wizard**.
- Click **Next** twice, and in the **GOSALES** data source, expand the **GOSALESRT** schema.
- Expand **Tables**, and select:
 - **RETAILER**
 - **RETAILER_SITE**
 - **RETAILER_TYPE**
- In **Generate Relationships**, select **Both**, and click **Import**.

Task 2. Organize the entity relationship diagram.

- In the center pane, in **Diagram**, click **Auto Layout**, and ensure that the **Layout Style** is set to **Standard**, and the distances are set to **30**.



This material is meant for IBM Academic Initiative use only. NOT FOR RESALE

© 2003, 2015, IBM Corporation

This guide contains proprietary information which is protected by copyright. No part of this document may be photocopied, reproduced, or translated into another language without a legal license agreement from IBM Corporation.

Task 3. Test that you have access to the new data.

- In the **Project Viewer**, in the **gosales** namespace, select the following query items:
 - RETAILER.COMPANY_NAME**
 - RETAILER_SITE.RTL_CITY**
 - RETAILER_TYPE.TYPE_NAME_EN**
- Right-click one of the selected items, click **Test**, and then click **Test Sample**.

The results appear similar to the following:

Test results		
COMPANY_NAME	RTL_CITY	TYPE_NAME_EN
Golf España	Logroño	Golf Shop
ActiForme	Paris	Equipment Rental Store
ActiForme	Strasbourg	Equipment Rental Store
ActiForme	Paris	Equipment Rental Store
SportsClub	Lyon	Golf Shop
SportsClub	Nice	Golf Shop
Anapuma	Orléans	Direct Marketing
Cordages Discount	Lyon	Warehouse Store
Cordages Discount	Paris	Warehouse Store
Cordages Discount	Paris	Warehouse Store
Cordages Discount	Nice	Warehouse Store

You have added the retailer information that your business analysts require to report on sales by retailer.

- In the **File** menu, click **Save**, and then close **Framework Manager**.



Preparing Reusable Metadata

IBM Cognos BI



Business Analytics software

© 2015 IBM Corporation

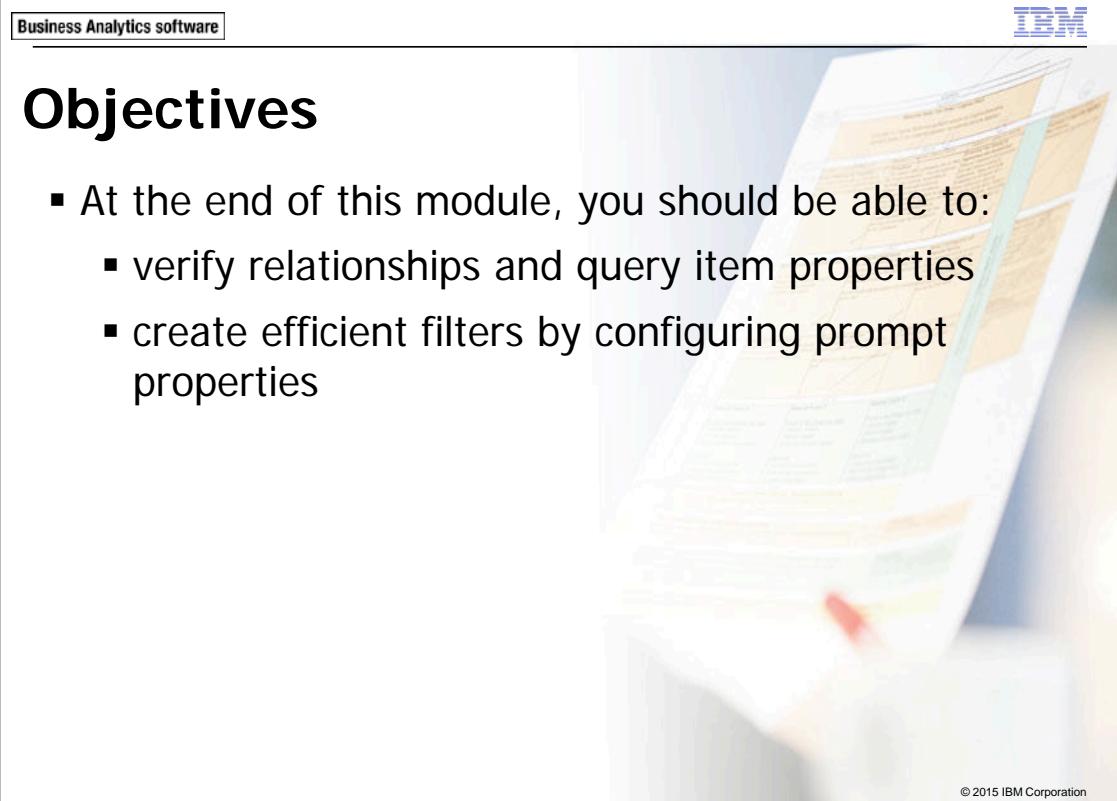
This material is meant for IBM Academic Initiative use only. NOT FOR RESALE

© 2003, 2015, IBM Corporation

This guide contains proprietary information which is protected by copyright. No part of this document may be photocopied, reproduced, or translated into another language without a legal license agreement from IBM Corporation.

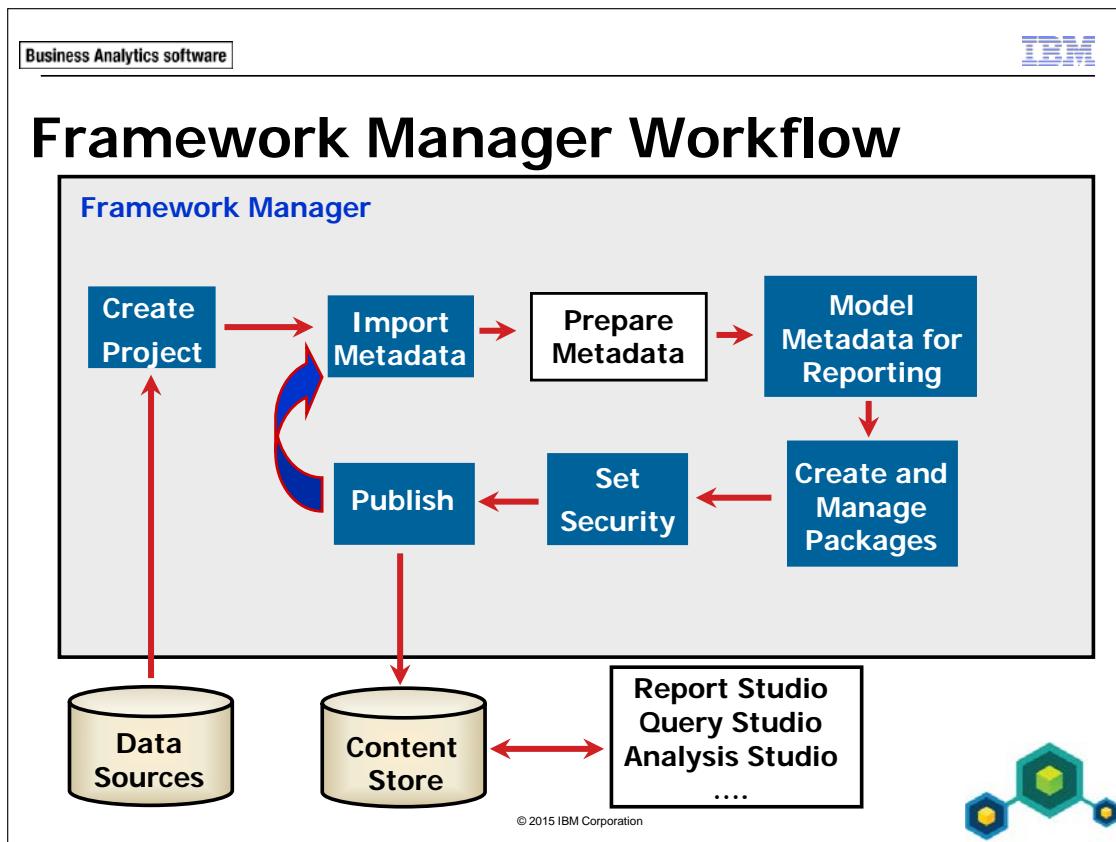
Objectives

- At the end of this module, you should be able to:
 - verify relationships and query item properties
 - create efficient filters by configuring prompt properties

A blurred background image of a person's hands interacting with a computer monitor. The monitor displays a complex, multi-layered data model or metadata structure with various nodes and connections, likely representing the IBM Business Analytics software mentioned in the header.

© 2015 IBM Corporation

The demos and workshops in this module represent the early phases of model development. You have gathered requirements and will now begin to design the metadata models that meet those requirements. The solutions in this module are not necessarily meant to be the final view seen by report authors. You test the results of the metadata as you design the model.



The Prepare Metadata phase involves verifying and modifying relationships and object properties and customizing metadata for run time.

These changes can be made either to the original data source query subjects (in the Foundation Objects View), or to the model query subjects (such as those you will create later in the course). Changes to the original data source query subjects are reusable. These query subjects may later be used in several different model query subjects, each of which will inherit the same changes. However, if you do not intend for a modification to be universal, then it should be made in the specific model query subjects that require that change.

Verifying Relationships

- Ensure your model has all required relationships after import.
- Ensure the join cardinality between query subjects meet your needs
 - 0..n - zero occurrences to multiple occurrences
 - 1..n - one occurrence to multiple occurrences
 - 0..1 - zero occurrences to one occurrence
 - 1..1 - must have one occurrence

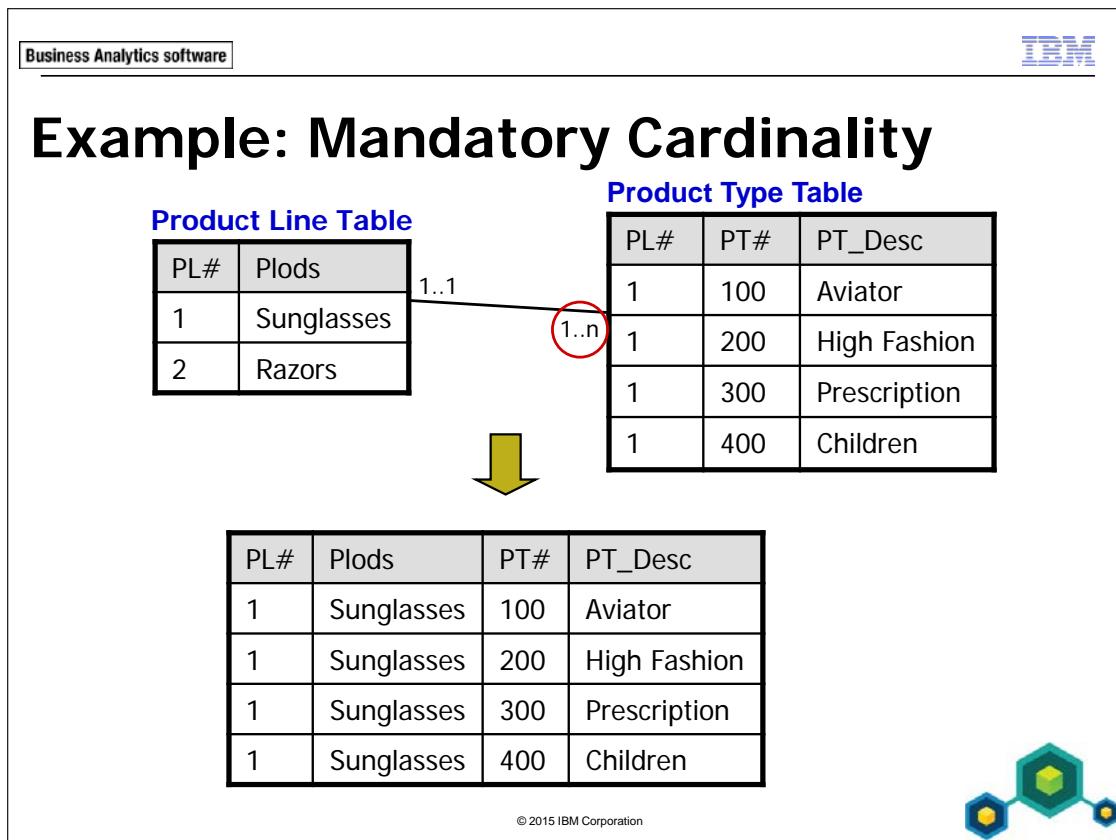
© 2015 IBM Corporation



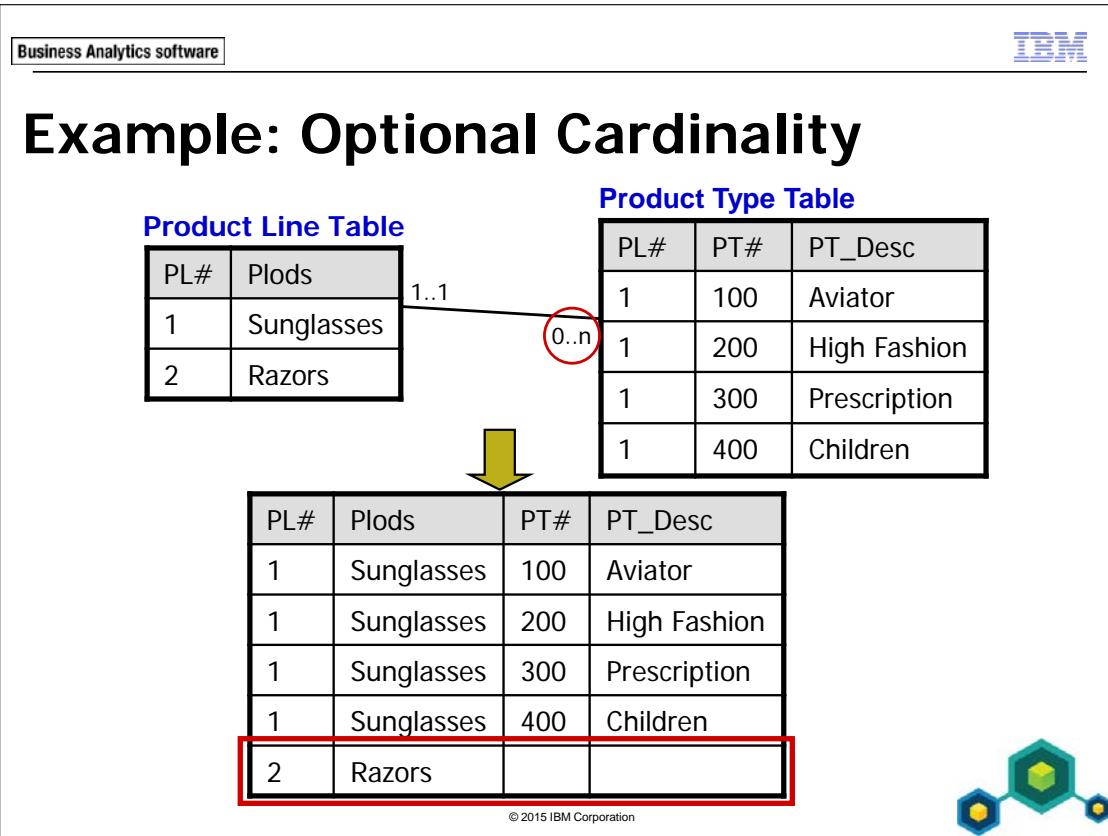
Relationships are maintained in the Object Diagram or Context Explorer.

When verifying your relationships, you must ensure that the appropriate relationships exist to meet your reporting needs and you must decide if you require optional or mandatory cardinalities. Optional cardinalities require more processing, but may be needed to return the desired results.

Optional cardinality is represented by a 0 as seen in the 0..n and 0..1 examples.



In the slide example, only rows in which the join condition is met are returned (where the product line number exists in both tables). The Razors product line is omitted from the record set.



Optional cardinality is represented by a 0 as seen in the 0..n in the above example.

With the cardinality set to 0..n on the Product Type table, all rows from the Product Line table are returned since a mandatory match in the Product Line table is not required. The Razors product line now appears in the record set.

In this course, we primarily use mandatory cardinalities (inner joins), as they provide better performance. However, if you require optional cardinality (outer join for reporting purposes, you can implement it, and optimize performance in other ways, such as adding filters, to offset the performance impact of outer joins.

Relationships are maintained in the Object Diagram or Context Explorer.

When verifying your relationships, you must ensure that the appropriate relationships exist to meet your reporting needs and you must decide if you require optional or mandatory cardinalities. Optional cardinalities require more processing, but may be needed to return the desired results.

Demo 1: Verify Relationships

Purpose:

To meet reporting requirements, you will verify your relationships to ensure none are missing and that the cardinality is set accordingly.

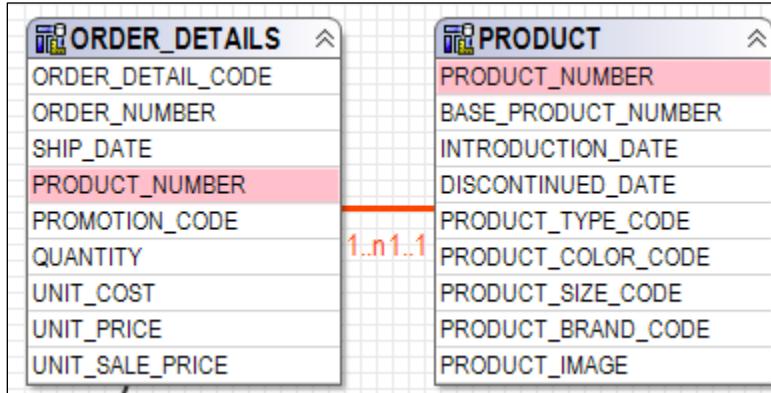
Component: Framework Manager

Project: GO Operational

Task 1. Examine relationships.

1. In Framework Manager, open the **GO Operational** project located at **C:\Edcognos\B5A52\CBIFM-Start Files\Module 5\GO Operational**. This is the baseline project created in the previous module. If prompted, log in as User ID **admin**, and Password **Education1**.
2. In the middle pane, click **Diagram**, and then double-click the **gosales** box to give it focus in the diagram.
3. Click the **Auto Layout** button, and set the horizontal and vertical distance to **25** to increase the distance between objects.
4. Click the relationship between **ORDER_DETAILS** and **PRODUCT**.

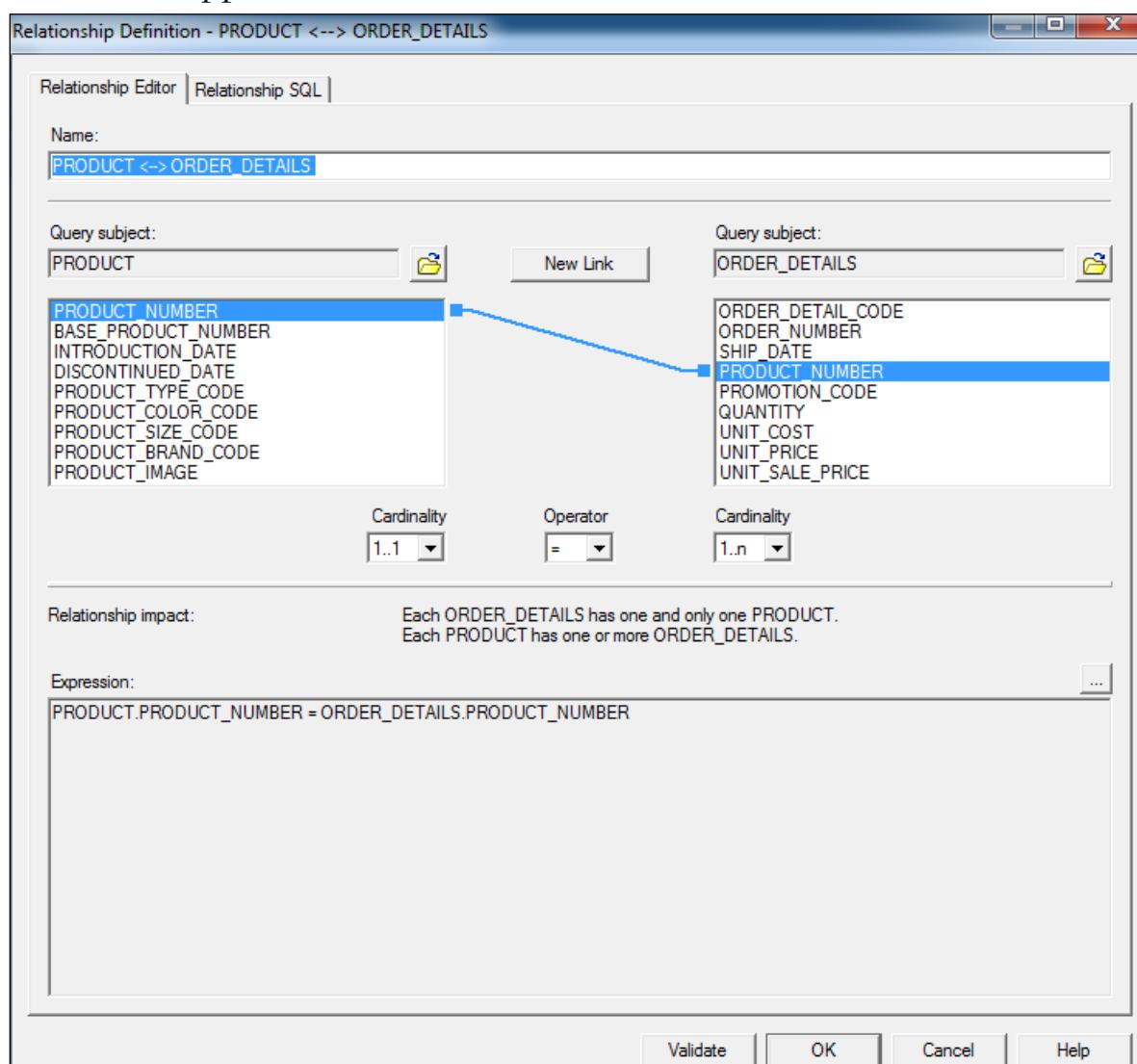
The results appear as follows:



The relationship line turns red and the query items used in the join are highlighted, in this case **PRODUCT_NUMBER**.

5. Double-click the relationship.

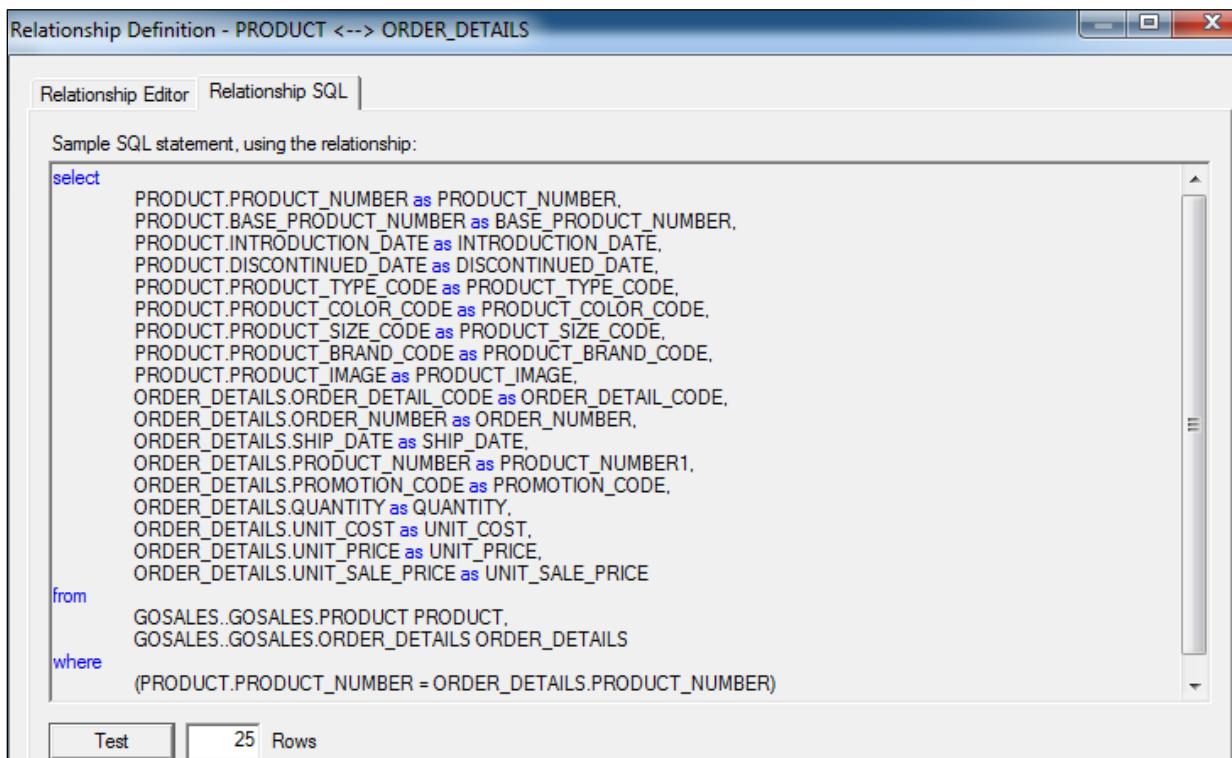
The results appear as follows:



You can change the query items used in the join, the cardinality settings, the operator, or edit the expression to create a more complex join.

6. Click the **Relationship SQL** tab.

The results appear as follows:



```

Relationship Definition - PRODUCT <--> ORDER_DETAILS

Relationship Editor Relationship SQL

Sample SQL statement, using the relationship:
select
    PRODUCT.PRODUCT_NUMBER as PRODUCT_NUMBER,
    PRODUCT.BASE_PRODUCT_NUMBER as BASE_PRODUCT_NUMBER,
    PRODUCT.INTRODUCTION_DATE as INTRODUCTION_DATE,
    PRODUCT.DISCONTINUED_DATE as DISCONTINUED_DATE,
    PRODUCT.PRODUCT_TYPE_CODE as PRODUCT_TYPE_CODE,
    PRODUCT.PRODUCT_COLOR_CODE as PRODUCT_COLOR_CODE,
    PRODUCT.PRODUCT_SIZE_CODE as PRODUCT_SIZE_CODE,
    PRODUCT.PRODUCT_BRAND_CODE as PRODUCT_BRAND_CODE,
    PRODUCT.PRODUCT_IMAGE as PRODUCT_IMAGE,
    ORDER_DETAILS.ORDER_DETAIL_CODE as ORDER_DETAIL_CODE,
    ORDER_DETAILS.ORDER_NUMBER as ORDER_NUMBER,
    ORDER_DETAILS.SHIP_DATE as SHIP_DATE,
    ORDER_DETAILS.PRODUCT_NUMBER as PRODUCT_NUMBER1,
    ORDER_DETAILS.PROMOTION_CODE as PROMOTION_CODE,
    ORDER_DETAILS.QUANTITY as QUANTITY,
    ORDER_DETAILS.UNIT_COST as UNIT_COST,
    ORDER_DETAILS.UNIT_PRICE as UNIT_PRICE,
    ORDER_DETAILS.UNIT_SALE_PRICE as UNIT_SALE_PRICE
from
    GOSALES..GOSALES.PRODUCT PRODUCT,
    GOSALES..GOSALES.ORDER_DETAILS ORDER_DETAILS
where
    (PRODUCT.PRODUCT_NUMBER = ORDER_DETAILS.PRODUCT_NUMBER)

Test 25 Rows

```

Here you can view the SQL that is generated to join these two query subjects together and can test the items returned by the join.

7. Click **Cancel**.

8. In the bottom right corner of the diagram, click **Diagram Overview** .

Tip: This icon only appears when there is more data than the window can display. If it is not visible, minimize the screen to the point where horizontal and vertical scroll bars appear.

9. Drag the red viewing window to view all parts of the diagram.

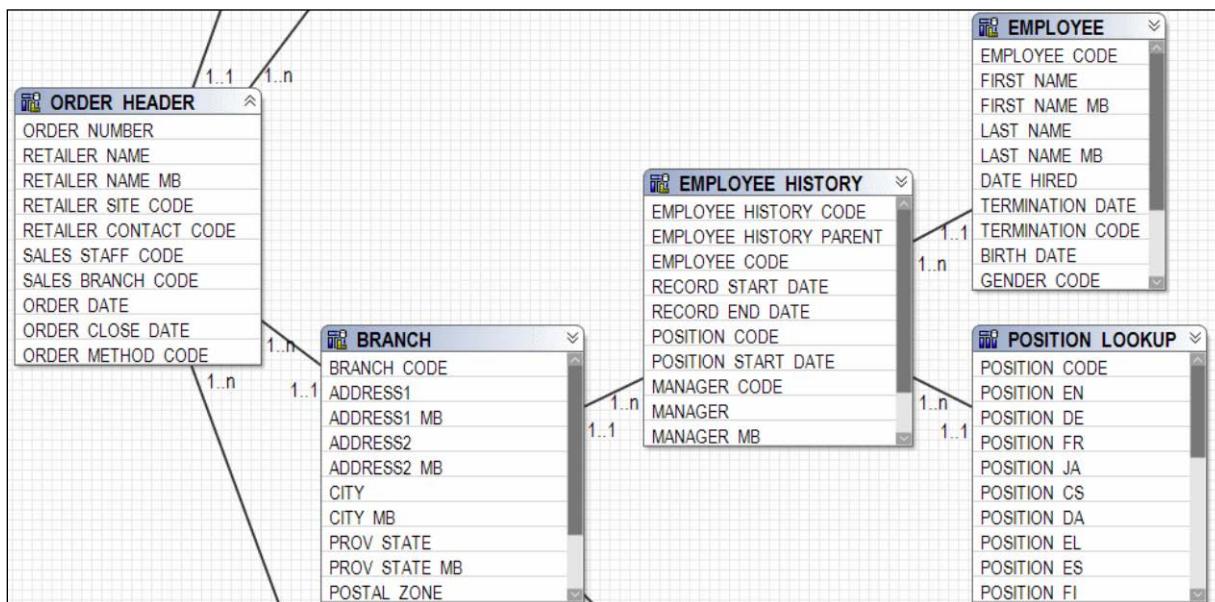
This feature is useful when dealing with larger models. You can see that all query subjects are linked to at least one other query subject, and that most relationships are one-to-many. You will examine one relationship in detail.

10. Close the **Diagram Navigation** window.

We recommend that you examine each relationship to verify that it is the relationship required for reporting purposes (with the appropriate cardinality) and make note of any missing relationships, where issues might occur, or items may need to be revisited.

Task 2. Identify missing relationships.

1. Examine the portion of the diagram below:



EMPLOYEE is joined to EMPLOYEE_HISTORY in a one-to-many relationship, allowing you to report on past positions. You will test this in the next task. BRANCH is also joined to EMPLOYEE_HISTORY in a one-to-many relationship. In this scenario EMPLOYEE_HISTORY is acting as a bridge table in a many-to-many relationship between BRANCH and EMPLOYEE. An employee may have worked in several branches and a branch may have several employees.

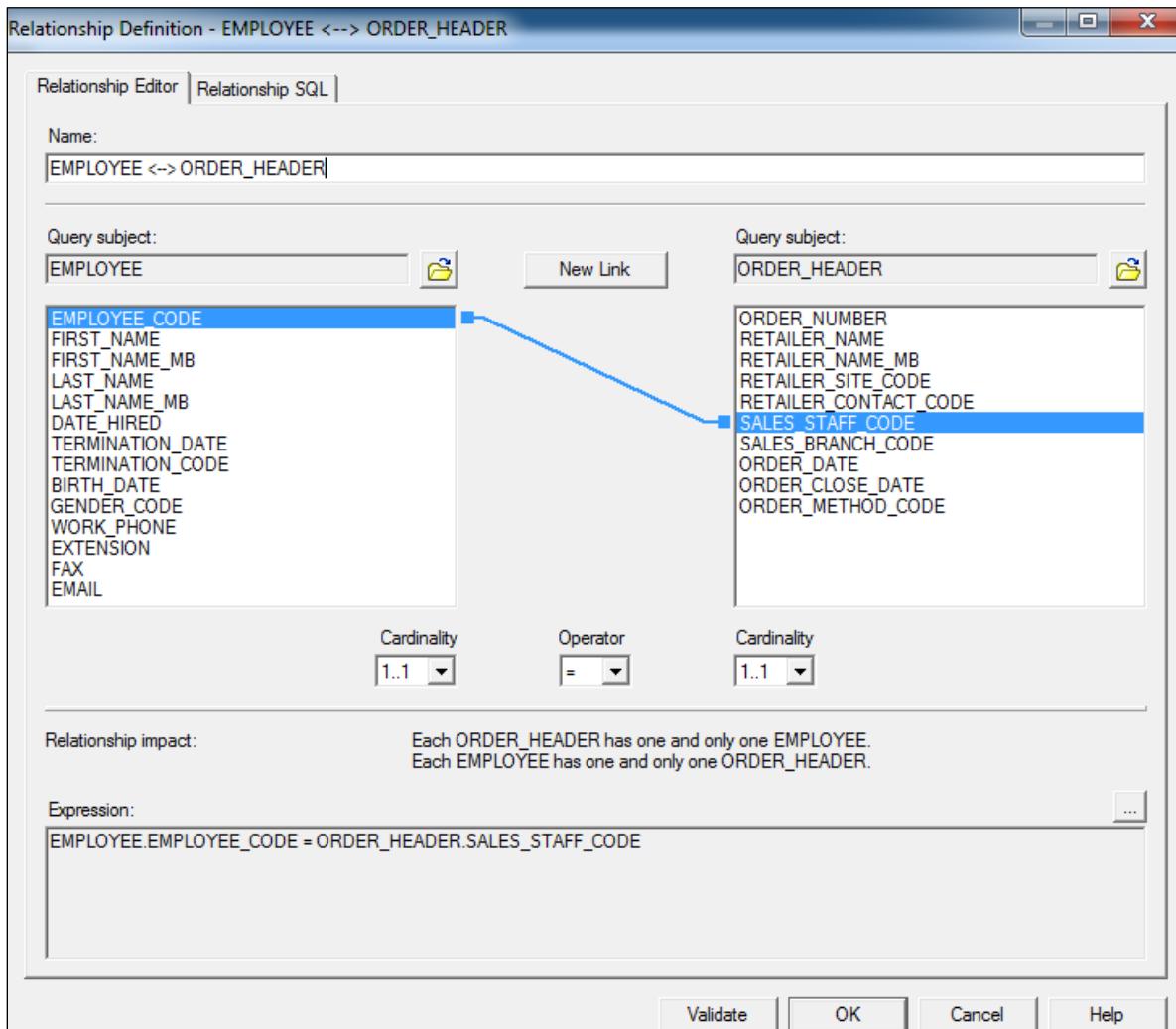
BRANCH has a relationship to ORDER_HEADER that indicates which branch made a sale. EMPLOYEE does not have a relationship to ORDER_HEADER. In this state, you cannot report on sales by staff, which is one of the requirements. EMPLOYEE needs a relationship to ORDER_HEADER.

In speaking with the database modeler, you discover that the relationship between the two objects should be EMPLOYEE.EMPLOYEE_CODE to ORDER_HEADER.SALES_STAFF_CODE. The names are different, but they do in fact represent the same information.

2. In the diagram, Ctrl+click **EMPLOYEE.EMPLOYEE_CODE** and **ORDER_HEADER.SALES_STAFF_CODE**.

3. Right-click one of the selected items, point to **Create**, and then click **Relationship**.

The results appear as follows:



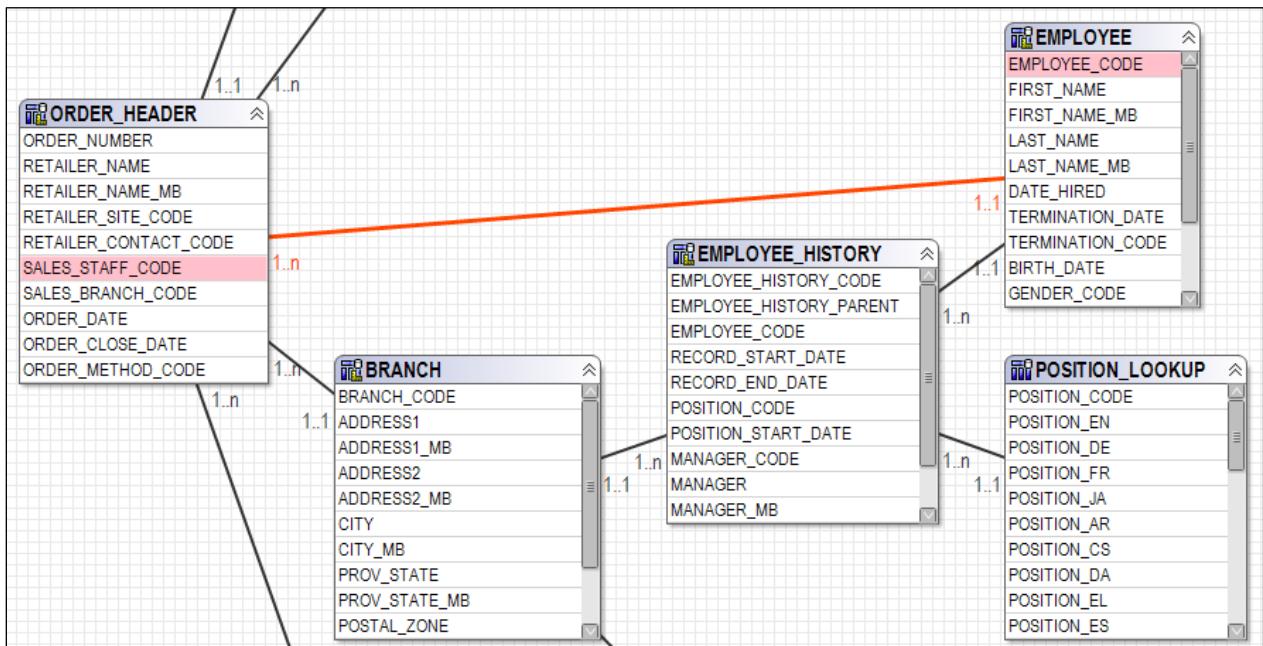
A relationship with the selected query items is created.

4. Under **ORDER_HEADER**, set the **Cardinality** to **1..n**.

This reflects the fact that each Employee can have one or more Order Headers.

- Click **Validate**, and then click **OK**.

The results appear as follows:



At this point, you are just ensuring that all required relationships are in place. Refining the model for presentation and predictable results will occur later in the modeling process.

Task 3. Test relationships.

You will now test one relationship to help you understand the data as well as test the new relationship you just created.

- Ctrl+click to select the following query items:

Query Subject	Query Item
EMPLOYEE	FIRST_NAME LAST_NAME
EMPLOYEE_HISTORY	RECORD_START_DATE RECORD_END_DATE MANAGER BRANCH_CODE
POSITION_LOOKUP	POSITION_EN

- Right-click one of the selected items, click **Test**, and then click **Test Sample**.

3. Scroll down until you see the manager named **Maria Iacobucci**.

The results appear as follows:

The screenshot shows a software interface for 'Test' and 'Query Information'. A checkbox labeled 'Auto Sum' is checked. The 'Test results' grid displays data with columns: FIRST_NAME, LAST_NAME, RECORD_START_DATE, and RECORD_END_DATE. There are two rows for 'Maria': one row where 'FIRST_NAME' is 'Maria' and 'LAST_NAME' is 'Iacobucci', and another row where 'FIRST_NAME' is 'Maria' and 'LAST_NAME' is 'Iacobucci'.

Test results			
FIRST_NAME	LAST_NAME	RECORD_START_DATE	RECORD_END_DATE
Gunter	Erler	Apr 18, 2013 12:00:00 AM	
Björn	Winkler	Jul 13, 2010 12:00:00 AM	Dec 19, 2010 12:00:00 AM
Björn	Winkler	Dec 20, 2010 12:00:00 AM	
Fritz	Hirsch	Dec 20, 2010 12:00:00 AM	
Jörg	Kunze	Feb 18, 2009 12:00:00 AM	
Silvano	Allessori	Jun 27, 2012 12:00:00 AM	
Mana	Iacobucci	Mar 25, 2008 12:00:00 AM	Apr 4, 2011 12:00:00 AM
Maria	Iacobucci	Apr 5, 2011 12:00:00 AM	
Alessandra	Torta	Dec 15, 2009 12:00:00 AM	
Belinda	Jansen-Velasquez	Aug 1, 2009 12:00:00 AM	

There are two rows of data for Maria because she changed managers (perhaps moving to a different part of the organization, but within the same branch).

4. Click **Close**.

You will now test the new relationship between EMPLOYEE and ORDER_HEADER.

5. In the diagram, select and **Test** the following query items:

Query Subject	Query Item
EMPLOYEE	FIRST_NAME LAST_NAME EMPLOYEE_CODE
ORDER_HEADER	SALES_STAFF_CODE
ORDER_DETAILS	QUANTITY

The results are similar to the following:

Test results				
FIRST_NAME	LAST_NAME	EMPLOYEE_CODE	SALES_STAFF_CODE	QUANTITY
Charles	Laurel	10798	10798	256
Charles	Laurel	10798	10798	92
Alexandre	Pereira	10406	10406	162
Alexandre	Pereira	10406	10406	172
Alexandre	Pereira	10406	10406	74
Alexandre	Pereira	10406	10406	90
Alexandre	Pereira	10406	10406	422
Alexandre	Pereira	10406	10406	3252
Alexandre	Pereira	10406	10406	1107
Alexandre	Pereira	10406	10406	88
Alexandre	Pereira	10406	10406	19

Since the EMPLOYEE_CODE values and SALES_STAFF_CODE values match, you can use Auto Sum to display how many items each employee sold overall.

6. Select **Auto Sum**, and then click **Test Sample** again.

The results appear as follows:

Test results				
FIRST_NAME	LAST_NAME	EMPLOYEE_CODE	SALES_STAFF_CODE	QUANTITY
Aaltje	Hansen	10359	10100025	470837
Abram	Ruiz	10237	12233215	862052
Adda	Heijman	10360	9531200	471856
Adriaantje	Haanraads	10361	8900099	535961
Agatha	Reyes	10307	1123463	610229
Agnelo	Chavez	10238	5661614	334984
Agnes	Ramos	10354	12600818	560574
Aidan	Chaplin	10471	628260	237059
Aiko	Watanabe	10572	2801580	849307
Aila	Forssell	10719	1404189	415057
Aimi	Tanaka	10571	708257	257727

Quantity is now summed for each employee. The relationship is working as expected. However, notice the SALES_STAFF_CODE value. It seems to have been summed as well. This is because it was set as an integer during import. You will examine and fix this issue in the next demo.

7. Click **Close**, click **Save**, and keep **Framework Manager** open for the next demo.

Results:

To meet reporting requirements, you verified your relationships and added a missing relationship.

Verifying Query Item Properties

- After import, verify that the metadata represents the business needs.
 - Should Usage for Manager_Code = Identifier, Fact, or Attribute?
 - Should Regular Aggregate for Unit_Price = SUM?
 - Should a prompt on Country Name retrieve through Country Code?

© 2015 IBM Corporation



Framework Manager populates object properties, such as Name, Usage, and Description, during import. When you see a property value that seems questionable, you should ask the database administrator how the item is configured in the database.

Facts are numeric or time-interval, non-indexed columns. Identifiers are key, index, date, datetime, or any indexed columns. Attributes are typically strings.

The Regular Aggregate property for numeric facts (measures) defaults to SUM, which is correct for most measures (such as REVENUE and QUANTITY). However a report identifying four tents sold to a customer would probably not display the sum of their prices, but instead the average.

Take advantage of Prompt properties to improve performance by causing automatic retrieval through indexes.

Demo 2: Verify and Modify Query Item Properties

Purpose:

To meet reporting requirements, you will verify that important query item properties, such as Usage and Regular Aggregate, are correct. You will also ensure efficient queries using filters by configuring query items to use an indexed key value rather than a non-indexed string value in the filter.

Components: Framework Manager, Query Studio, Report Studio

Project: GO Operational

Package: GO Operational

Task 1. Verify Usage property settings.

1. If necessary, from the **View** menu, click **Properties**.
2. In the **Project Viewer**, expand **Foundation Object View > gosales > ORDER_HEADER**, and then click **ORDER_NUMBER**.

Both the query item icon  and the Usage property field tell you that ORDER_NUMBER is an identifier, and were identified as such during import since this is the primary key in the database table.

3. Click **SALES_STAFF_CODE**.

In comparison, SALES_STAFF_CODE has a measure icon (also known as a Fact)  and its Usage property is set to Fact. This was specified during the import because it is a non-indexed numeric field in the database.

You know that this query item is not a measure. It is used for associating several employees to an order. You will set this query item's Usage property to Attribute. If you know that any item would be used in relationships or that report authors would filter on an item regularly, you would ask the database administrator to index the field in the database. At that point you would set the Usage property to Identifier. For now, you will set SALES_STAFF_CODE as Attribute to indicate to yourself and other modelers that this integer field is not indexed in the database.

4. In the **Properties** pane, click the **Usage** property, and then select **Attribute** from the list.
You will now change multiple properties at once to correct other items that Framework Manager incorrectly set as facts.
5. Under **ORDER_HEADER**, Ctl+click to select the following items:
 - **RETAILER_SITE_CODE**
 - **RETAILER_CONTACT_CODE**
 - **SALES_BRANCH_CODE**
6. In the **Properties** pane, in the **RETAILER_SITE_CODE** row, scroll to the right, click **Fact** below the **Usage** heading, and then select **Attribute** from the list.
7. Drag the small black arrow beneath the value downward to change the values to **Attribute** for the other two query items.

Task 2. Use Bulk Replace to change property settings.

Rather than go through each query subject and manually change the Usage setting for codes from Fact to Attribute, you can use the Search tool to perform bulk changes.

1. If necessary, from the **View** menu, click **Tools**.
2. In the **Tools** pane, click the **Search** tab, and then in the **Search string** box, type **_CODE**.
3. Beside the **Search** button, click  to see additional search options.
4. Set the following:
 - Class: **Query Item**
 - Property: **Object Name**
5. Click **Search**, and then resize the **Tools** pane to view the results.
You will search a subset of this result set to display only items whose Usage property is set to Fact.
6. Select the **Subset** check box, in the **Search string** box, type **fact**, and then change the **Property** selection to **Usage**.

7. Click Search.

The results appear as follows:

Object	Property	Value
PROMOTION_CODE	GO Operational Model \ Foundation...	fact
PRODUCT_COLOR_CODE	GO Operational Model \ Foundation...	fact
PRODUCT_SIZE_CODE	GO Operational Model \ Foundation...	fact
PRODUCT_BRAND_CODE	GO Operational Model \ Foundation...	fact
WAREHOUSE_BRANCH_CODE	GO Operational Model \ Foundation...	fact
TERMINATION_CODE	GO Operational Model \ Foundation...	fact
GENDER_CODE	GO Operational Model \ Foundation...	fact
EMPLOYEE_HISTORY_CODE	GO Operational Model \ Foundation...	fact
MANAGER_CODE	GO Operational Model \ Foundation...	fact
RTL_ACTIVITY_STATUS_CO...	GO Operational Model \ Foundation...	fact

You will now use the Bulk Replace feature to change the Usage property for these items to Attribute.

8. In the bottom-right corner, click **Bulk Replace**.
9. Set **Replace with** to **Attribute**.
10. Click **Replace All**, and then click **Yes**.

The results appear as follows:

Object	Property	Value
PROMOTION_CODE	GO Operational Model \ Foundation...	attribute
PRODUCT_COLOR_CODE	GO Operational Model \ Foundation...	attribute
PRODUCT_SIZE_CODE	GO Operational Model \ Foundation...	attribute
PRODUCT_BRAND_CODE	GO Operational Model \ Foundation...	attribute
WAREHOUSE_BRANCH_CODE	GO Operational Model \ Foundation...	attribute
TERMINATION_CODE	GO Operational Model \ Foundation...	attribute
GENDER_CODE	GO Operational Model \ Foundation...	attribute
EMPLOYEE_HISTORY_CODE	GO Operational Model \ Foundation...	attribute
MANAGER_CODE	GO Operational Model \ Foundation...	attribute
RTL_ACTIVITY_STATUS_CO...	GO Operational Model \ Foundation...	attribute

All Usage properties are now changed to attribute.

Using the Search tool can save a lot of time when performing these types of tasks. However, you should still examine all query items to ensure the settings are correct. For example, BASE_PRODUCT_NUMBER from PRODUCT and EMPLOYEE_HISTORY_PARENT from EMPLOYEE_HISTORY are still identified as facts. The search did not catch these since they did not contain the string _CODE.

Task 3. Verify Regular Aggregate property settings.

- Select and **Test** the following query items with **Auto Sum** enabled:

Query Subject	Query Item
PRODUCT_LINE	PRODUCT_LINE_EN
PRODUCT_TYPE	PRODUCT_TYPE_EN
PRODUCT	PRODUCT_NUMBER
ORDER_DETAILS	QUANTITY UNIT_SALE_PRICE

The results appear as follows:

Test results				
PRODUCT_LINE_EN	PRODUCT_TYPE_EN	PRODUCT_NUMBER	QUANTITY	UNIT_SALE_PRICE
Camping Equipment	Cooking Gear	1110	4308828	6832.99
Camping Equipment	Cooking Gear	2110	965723	14111.63
Camping Equipment	Cooking Gear	3110	866669	26852.57
Camping Equipment	Cooking Gear	4110	1812123	3966
Camping Equipment	Cooking Gear	5110	813780	62344.6
Camping Equipment	Cooking Gear	6110	442136	144169.88
Camping Equipment	Cooking Gear	7110	686493	74157.07
Camping Equipment	Cooking Gear	8110	245559	190834.99
Camping Equipment	Cooking Gear	9110	2336950	13757.15
Camping Equipment	Cooking Gear	10110	922090	21345.53
Camping Equipment	Tents	11110	477692	529801.2

Notice UNIT_SALE_PRICE is summed. This does not meet the reporting requirements. In a report on ORDER_DETAILS rolled up for each product you would expect the QUANTITY to be summed, but it would make no sense to sum up the UNIT_SALE_PRICE (the sale price for one unit of each product in the order), since you may have sold several units in one order detail. You will change this aggregation to default to something more useful. In this case, Average. The same situation applies to UNIT_COST and UNIT_PRICE. (Later in the course, you will create a Revenue calculation, QUANTITY * UNIT_SALE_PRICE.)

- Click **Close**, and then in the **Project Viewer**, under **ORDER_DETAILS**, select the following items:

UNIT_COST

UNIT_PRICE

UNIT_SALE_PRICE

- In the **Properties** pane, change the **Regular Aggregate** property values to **Average** for all the items.

The results appear as follows:

Properties								
Properties		Language						
Item Name	Size	Is Nullable	Display Type	MIME Type	Prompt Info	Regular Aggregate	Scripted	Scripted Value
UNIT_COST	10	True	Value			Average	S	
UNIT_PRICE	10	True	Value			Average	S	
UNIT_SALE_PRICE	10	True	Value			Average	S	

You will change the format for these objects to \$USD Currency.

- In the **UNIT_COST** row, under **Format**, click <Click to edit.>.
- Under **Format type**, select **Currency**, and then in the **Properties** pane, set **Currency** to **\$(USD) - United States of America, dollar**.
- Click **OK**, and under **Format**, drag the small **black arrow** down to change the property setting for the other two items.
- Click **OK**.

The results appear as follows:

Properties								
Properties		Language						
Item Name	Internal Name	Is Hidden	Usage	Format	Currency	Data Type	Precision	Scripted
UNIT_COST	UNIT_COST	False	Fact	<Click to edit.>	USD	Decimal	19	
UNIT_PRICE	UNIT_PRICE	False	Fact	<Click to edit.>	USD	Decimal	19	
UNIT_SALE_PRICE	UNIT_SALE_PRICE	False	Fact	<Click to edit.>	USD	Decimal	19	

- Save the project.

Task 4. Test property changes in Framework Manager.

1. Select and **Test** the following query items with **Auto Sum** enabled:

Query Subject	Query Item
PRODUCT_LINE	PRODUCT_LINE_EN
PRODUCT_TYPE	PRODUCT_TYPE_EN
PRODUCT	PRODUCT_NUMBER
ORDER_DETAILS	QUANTITY UNIT_SALE_PRICE

The results appear as follows:

Test results				
PRODUCT_LINE_EN	PRODUCT_TYPE_EN	PRODUCT_NUMBER	QUANTITY	UNIT_SALE_PRICE
Camping Equipment	Cooking Gear	1110	4308828	5.79066949152542
Camping Equipment	Cooking Gear	2110	965723	11.95900847457627
Camping Equipment	Cooking Gear	3110	866669	22.75641525423728
Camping Equipment	Cooking Gear	4110	1812123	3.36101694915254
Camping Equipment	Cooking Gear	5110	813780	44.7235294117647
Camping Equipment	Cooking Gear	6110	11107070502424514	

QUANTITY has been summed while UNIT_SALE_PRICE has been averaged. This is based on the property setting you specified earlier.

2. Click **Close**.

Task 5. Adjust Prompt properties.

Prompts can be either manually created by the report author, or generated when the report author filters an item or includes a prompt value (i.e. ?Product Line?) in an expression. The prompt type for each query item is defined in Framework Manager.

1. In the **Project Viewer**, under **PRODUCT_LINE** select **PRODUCT_LINE_EN**.
2. In the **Properties** pane, expand **Prompt Info**.

3. Click the **Prompt Type** value, and click the arrow to view the list of options.
The results appear as follows:

Prompt Info	
Prompt Type	Server Determined
Cascade On Item Reference	Server Determined
Display Item Reference	Edit Box Select Date Select Date/Time Select Interval Select Time Select Value
Filter Item Reference	Select with Search
Use Item Reference	Select with Tree
Regular Aggregate	
Semi-Aggregate	

You can specify the type of prompt that you want generated in Query Studio or the default prompt type for Report Studio. The default is for the server to determine the prompt type based on data type.

Here is a brief description of the other properties:

- **Cascade on Item Reference** is for cascading prompts in Report Studio, for example, where the list of Product Type choices is restricted to those within a selected Product Line.
- **Display Item Reference** identifies the default value that a manually created Report Studio prompt will display for a particular query item. For example, to see a list of PRODUCT_LINE_EN names when using PRODUCT_LINE_CODE in a prompt, set the Display Item Reference property for PRODUCT_LINE_CODE to PRODUCT_LINE_EN. If this field is left blank, it will default to the values returned by query item to which it belongs.
- **Use Item Reference** identifies the default value that a manually created Report Studio prompt will use in the query's filter. For example, for a Report Studio generated prompt on PRODUCT_LINE_EN to display a list of PRODUCT_LINE_EN names but retrieve data through the PRODUCT_LINE_CODE, set the Use Item Reference property in PRODUCT_LINE_EN to PRODUCT_LINE_CODE.
- **Filter Item Reference** identifies the value that an IBM Cognos 10 generated prompt will use to filter your query. For example, for Query Studio to display PRODUCT_LINE_EN values but use PRODUCT_LINE_CODE in the query's filter, set the Filter Item Reference property to PRODUCT_LINE_CODE.

You will set prompt properties on PRODUCT_LINE_EN and PRODUCT_TYPE_EN so that they use appropriate indexes in any prompt situation to retrieve values from the database. To achieve this, you will set the Use Item Reference property to ensure user generated prompts in Report Studio are efficient, and you will set the Filter Item Reference property to ensure any generic IBM Cognos generated prompt is efficient. You will also configure PRODUCT_TYPE_EN to cascade off of PRODUCT_LINE_EN.

4. Click the **Use Item Reference** value, and then click the **ellipsis (...)**.
5. Expand **Foundation Objects View > gosales > PRODUCT_LINE**, click **PRODUCT_LINE_CODE**, and then click **OK**.
6. Click the **Filter Item Reference** value, and click the **ellipsis (...)**.
7. Expand **Foundation Objects View > gosales > PRODUCT_LINE**, click **PRODUCT_LINE_CODE**, and then click **OK**.

The results appear as follows:

Prompt Info	
Prompt Type	Server Determined
Cascade On Item Reference	
Display Item Reference	
Filter Item Reference	[gosales].[PRODUCT_LINE].[PRODUCT_LINE_CODE]
Use Item Reference	[gosales].[PRODUCT_LINE].[PRODUCT_LINE_CODE]

8. Save the project.

Task 6. Test the prompt property change in Query Studio and Report Studio.

1. Publish the **GO Operational** package, overwriting the earlier version.
2. In **IBM Cognos Connection**, launch **Query Studio**, and select the **GO Operational** package.
3. In the **Insert Data** pane, expand **Foundation Objects View > gosales**, and then add the following items to the report:

Query Subject	Query Item
PRODUCT_LINE	PRODUCT_LINE_EN
ORDER_DETAILS	QUANTITY

4. Select the **PRODUCT_LINE_EN** column heading, and then click **Filter** .

5. Select **Camping Equipment** and then click **OK**.

The query retrieves the expected row.

PRODUCT_LINE_EN	QUANTITY
Camping Equipment	27,301,149
Summary	27,301,149

To view the underlying SQL, you will convert the Query Studio report to a Report Studio report.

6. Under **Menu**, click **Manage File**, and then click **Open in Report Studio**.
7. In **Report Studio**, from the **Tools** menu, click **Show Generated SQL/MDX**, and then, in the **Generated SQL/MDX** list, select **IBM Cognos SQL**.
Note that the Where clause uses
(PRODUCT_LINE.PRODUCT_LINE_CODE in (991)). This shows that you are retrieving data based on indexed codes rather than the less-efficient and non-indexed PRODUCT_LINE_EN strings.
8. Click **Close**, and then close **Report Studio** and **Query Studio** without saving the reports.

Results:

To meet reporting requirements, you verified that important query item properties such as Usage and Regular Aggregate were correct. Also, to make filtering more efficient, you configured PRODUCT_LINE_EN to use PRODUCT_LINE_CODE when filtering on PRODUCT_LINE_EN.

Summary

- You should now be able to:
 - verify relationships and query item properties
 - create efficient filters by configuring prompt properties

© 2015 IBM Corporation

This material is meant for IBM Academic Initiative use only. NOT FOR RESALE

© 2003, 2015, IBM Corporation

This guide contains proprietary information which is protected by copyright. No part of this document may be photocopied, reproduced, or translated into another language without a legal license agreement from IBM Corporation.

5-27

Workshop 1: Verify and Modify Query Item Properties

After every import of metadata, you need to verify that query item properties meet the report user requirements. To do so:

- Verify query item properties. Change the Usage property from Fact to Attribute where numeric query items are neither facts nor indexes.
- In the COUNTRY query subject, change COUNTRY_EN so that when a prompt is manually created or automatically generated it will cause retrieval through the COUNTRY_CODE key.
- Publish the package and verify your prompt works using Query Studio and Report Studio.
- In Framework Manager, save the project, close Framework Manager.
- Close all browser windows without saving the report.

For more information about where to work and the workshop results, refer to the Tasks and Results section that follows. If you need more information to complete a task, refer to earlier demos for detailed steps.

Workshop 1: Tasks and Results

Task 1. Change query item Usage properties.

- In the **Project Viewer**, in **Foundation Objects View > gosales**, Ctrl+click the following query items:
 - **PRODUCT > BASE_PRODUCT_NUMBER**
 - **EMPLOYEE_HISTORY > EMPLOYEE_HISTORY_PARENT**
- In the **Property** pane, change **Usage** to **Attribute** for the above numeric query items that are not facts.

Task 2. Change prompt properties.

- In the **Project Viewer**, select **COUNTRY > COUNTRY_EN**.
- In the **Property** pane, expand **Prompt Info**, and change the **Filter Item Reference** and **Use Item Reference** properties to **COUNTRY > COUNTRY_CODE**.

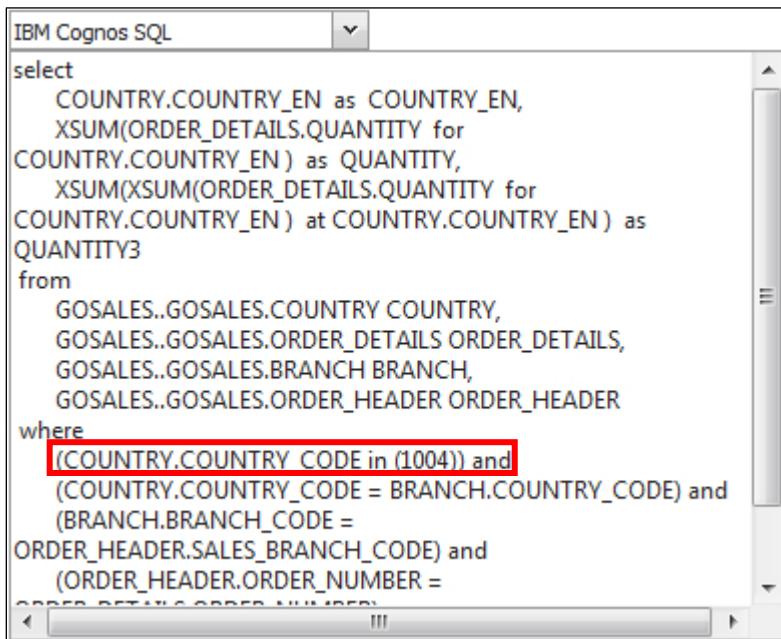
Task 3. Publish the package and verify that the prompt works using Query and Report Studio.

- Publish the **GO Operational** package.
- In **Query Studio**, use the **GO Operational** package to create a query using the following items:
 - **COUNTRY > COUNTRY_EN**
 - **ORDER_DETAILS > QUANTITY**
- Create a **Filter** on **COUNTRY_EN**, and set the value to **Canada**.

COUNTRY_EN	QUANTITY
Canada	4,052,045
Summary	4,052,045

- Open the report in **Report Studio** and use **Tools > Show Generated SQL/MDX** to verify that the data is filtered by **COUNTRY_CODE**.

The results appear as follows:



```
IBM Cognos SQL
select
    COUNTRY.COUNTRY_EN as COUNTRY_EN,
    XSUM(ORDER_DETAILS.QUANTITY for
COUNTRY.COUNTRY_EN) as QUANTITY,
    XSUM(XSUM(ORDER_DETAILS.QUANTITY for
COUNTRY.COUNTRY_EN) at COUNTRY.COUNTRY_EN) as
QUANTITY3
from
    GOSALES..GOSALES.COUNTRY COUNTRY,
    GOSALES..GOSALES.ORDER_DETAILS ORDER_DETAILS,
    GOSALES..GOSALES.BRANCH BRANCH,
    GOSALES..GOSALES.ORDER_HEADER ORDER_HEADER
where
    (COUNTRY.COUNTRY_CODE in (1004)) and
    (COUNTRY.COUNTRY_CODE = BRANCH.COUNTRY_CODE) and
    (BRANCH.BRANCH_CODE =
ORDER_HEADER.SALES_BRANCH_CODE) and
    (ORDER_HEADER.ORDER_NUMBER =
ORDER_DETAILS.ORDER_NUMBER)
```

The IBM Cognos SQL generated in Report Studio for the Query Studio query shows that you are accessing the data through **COUNTRY_CODE**.

- **Save** and **Close** the project in **Framework Manager**.
- Close all browser windows without saving the report, and then close Framework Manager.



Modeling for Predictable Results: Identify Reporting Issues

IBM Cognos BI

Business Analytics software

© 2015 IBM Corporation



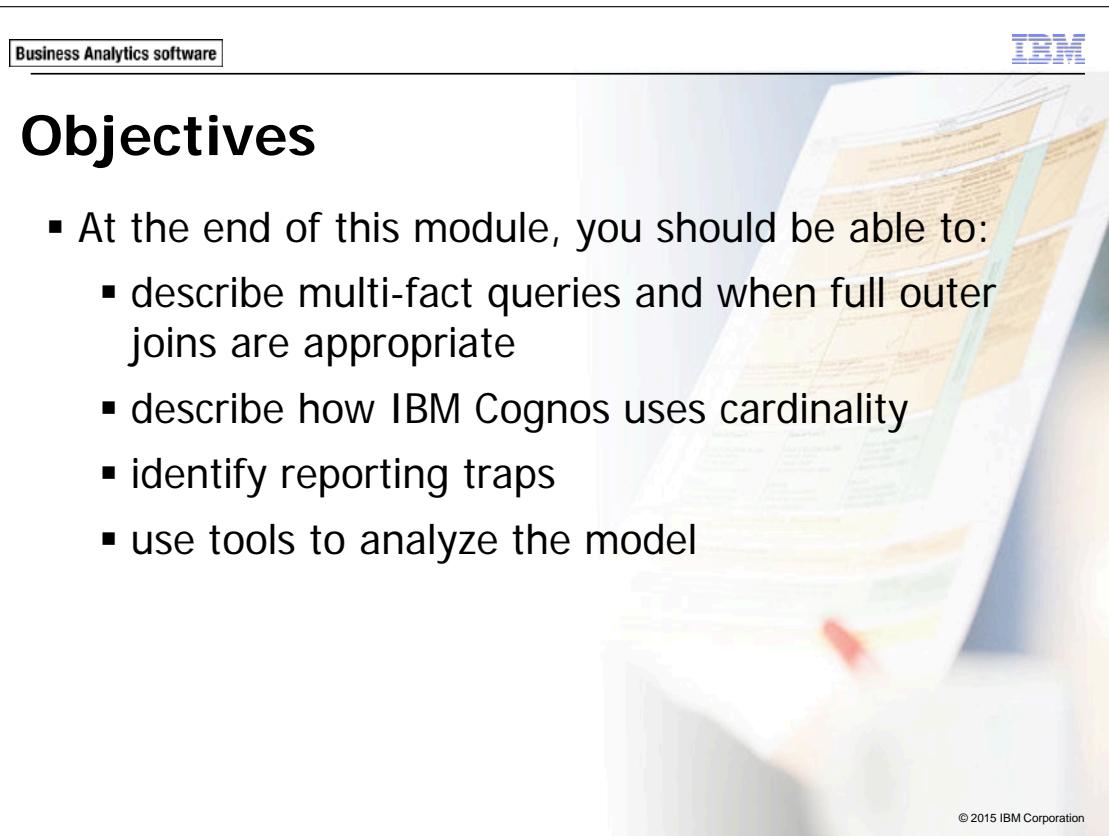
This material is meant for IBM Academic Initiative use only. NOT FOR RESALE

© 2003, 2015, IBM Corporation

This guide contains proprietary information which is protected by copyright. No part of this document may be photocopied, reproduced, or translated into another language without a legal license agreement from IBM Corporation.

Business Analytics software

IBM



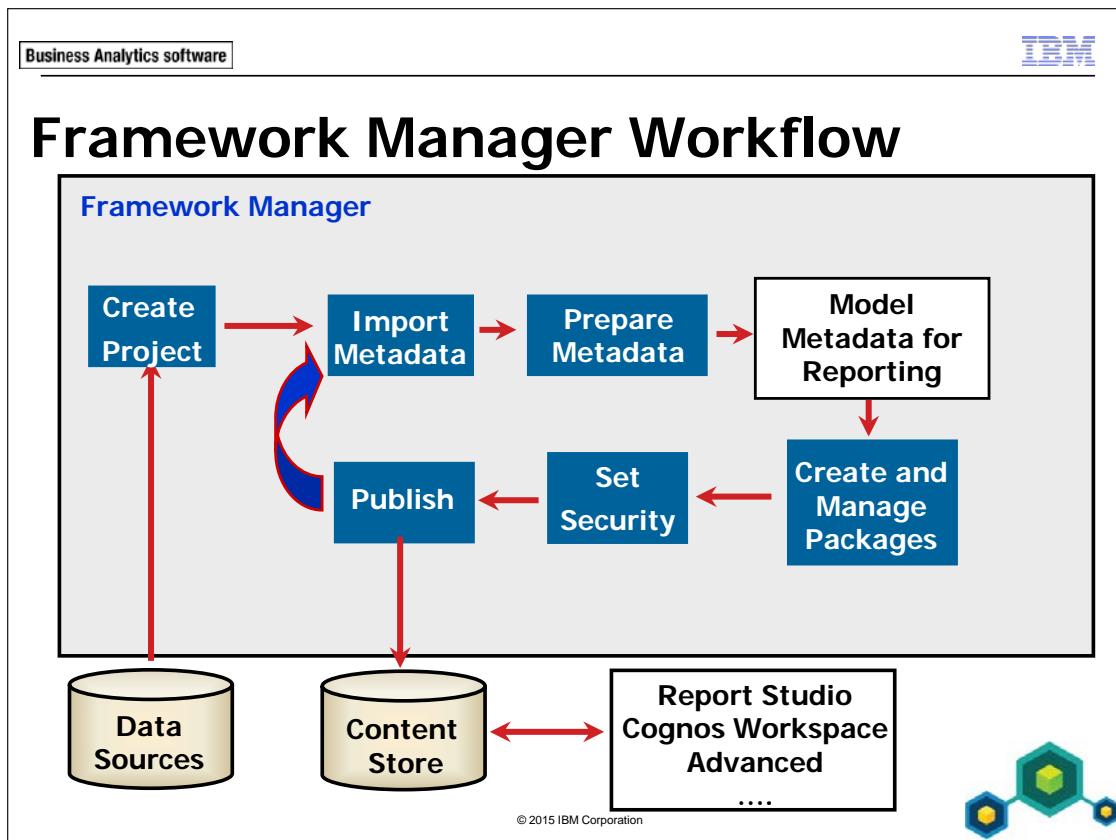
© 2015 IBM Corporation

Objectives

- At the end of this module, you should be able to:
 - describe multi-fact queries and when full outer joins are appropriate
 - describe how IBM Cognos uses cardinality
 - identify reporting traps
 - use tools to analyze the model

Unless otherwise specified in demo or workshop steps, you will always log on to IBM Cognos BI in the Local LDAP namespace using the following credentials:

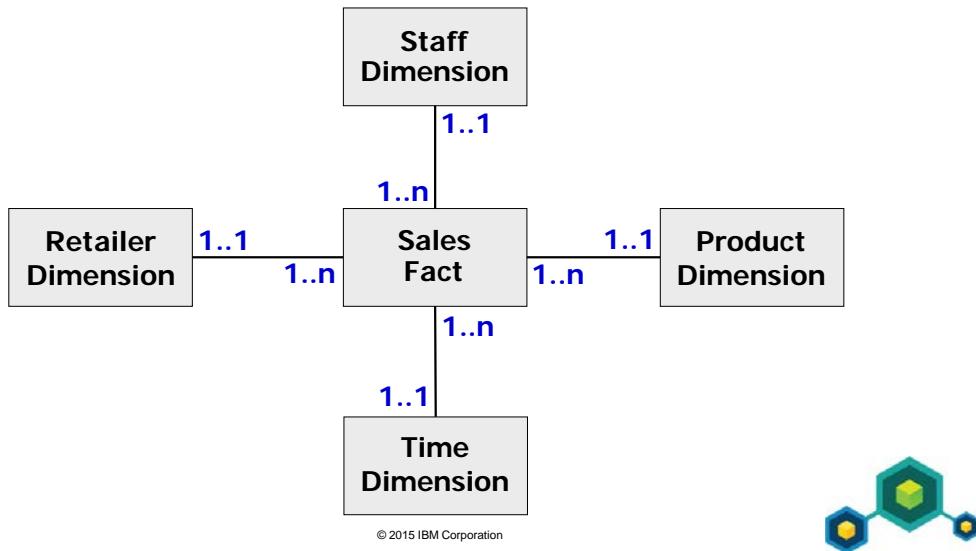
- User ID: admin
- Password: Education1



This module deals with identifying potential reporting issues before beginning to model the metadata. Solutions to the issues found in this module will be applied in later modules.

Single-Fact Queries

- A single-fact query consists of one or more measures from the same table



In the above example, if you only queried items from the dimensions, such as Product and Retailer, you would go through the fact table. Performance may be an issue if the fact table is very large. This is something to be aware of when viewing the generated SQL and considering performance for those types of queries.

If we use optional cardinalities, the full outer joins ensure that all data is returned, not just where matches occur. For example, if we wanted to see all sales, regardless of whether a sales rep made the sale, and see all sales reps, regardless of whether they made sales or not, we would set optional cardinality on both Staff Dimension and Sales Fact. This would generate a full outer join, ensuring that all rows are returned from each table. These types of scenarios should be configured with caution as they have the potential to be resource intensive. The measures can be constricted by dimensional attributes taken from related dimension query subjects at the 1..1 end of 1..1 to 1..n relationships.

In most cases, single-fact queries are not a problem for reporting and usually generate simple and expected SQL.

Business Analytics software

IBM

Multi-Fact Query Issues (1 of 3)

- Examine the following data:

DAY_DATE	ORDER_METHOD	SALE_TOTAL
01/01/2005	E-mail	\$10
01/02/2005	Telephone	\$25
01/03/2005	Web	\$40
01/04/2005	E-mail	\$20

Sales Fact

DAY_DATE	ORDER_METHOD	RETURN_QUANTITY
01/01/2005	E-mail	2
01/02/2005	Telephone	4
01/10/2005	Fax	15
01/11/2005	Sales visit	1

Returned Items Fact

Common information for
DAY_DATE and ORDER_METHOD

© 2015 IBM Corporation



While the first two rows have identical DAY_DATE and ORDER_METHOD information, rows three and four do not.

Multi-Fact Query Issues (2 of 3)

- an inner join in a select statement assumes that the same information exist in both facts

DAY_DATE	ORDER_METHOD	SALE_TOTAL
01/01/2005	E-mail	\$10
01/02/2005	Telephone	\$25
01/03/2005	Web	\$40
01/04/2005	E-mail	\$20

Sales Fact

DAY_DATE	ORDER_METHOD	RETURN_QUANTITY
01/01/2005	E-mail	2
01/02/2005	Telephone	4
01/10/2005	Fax	15
01/11/2005	Sales visit	1

Returned Items Fact

```
Select a.SALE_TOTAL, b.RETURN_QUANTITY from Sales a, Returns b
Where a.DAY_DATE = b. DAY_DATE and a.ORDER_METHOD = b.ORDER_METHOD
```

© 2015 IBM Corporation



An inner join would only return the first two records. If a filter ORDER_METHOD = 'Web' was added, 0 records would be returned.

Data is lost in this scenario.

Business Analytics software

IBM

Multi-Fact Query Issues (3 of 3)

- full outer joins can break queries into multiple selects, one for each fact table, and then merge the data

Sales Fact		
DAY_DATE	ORDER_METHOD	SALE_TOTAL
01/01/2005	E-mail	\$10
01/02/2005	Telephone	\$25
01/03/2005	Web	\$40
01/04/2005	E-mail	\$20

Returned Items Fact		
DAY_DATE	ORDER_METHOD	RETURN_QUANTITY
01/01/2005	E-mail	2
01/02/2005	Telephone	4
01/10/2005	Fax	15
01/11/2005	Sales visit	1

Report Set

DAY_DATE	ORDER_METHOD	SALE_TOTAL	RETURN_QUANTITY
01/01/2005	E-mail	\$10	2
01/02/2005	Telephone	\$25	4
01/03/2005	Web	\$40	
01/04/2005	E-mail	\$20	
01/10/2005	Fax		15
01/11/2005	Sales visit		1

© 2015 IBM Corporation



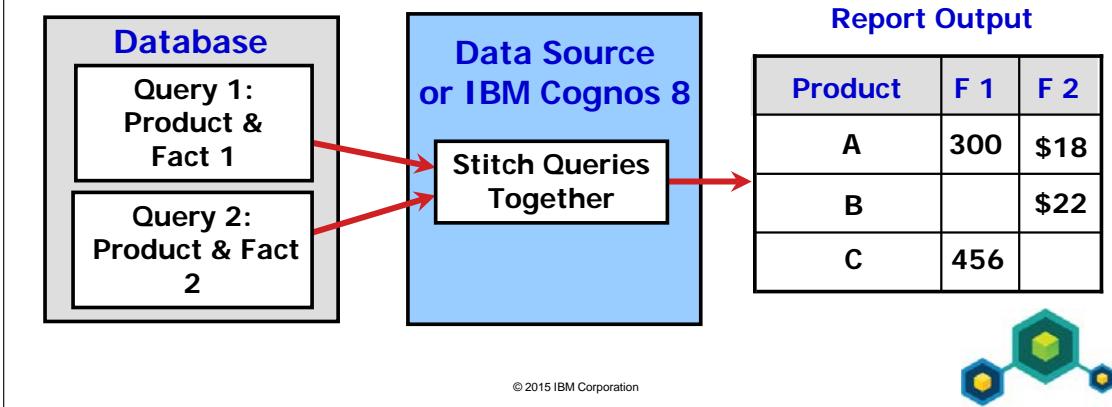
In the above example, no data is lost. This is what IBM Cognos refers to as a stitch query.

This type of data pattern is not exclusive to star schema data sources, it is also found in operational systems. For example, an employee has skills and also has billings. Employee is a conformed dimension that relates to a Skills table as well as a Billings table. To avoid losing any skills or billings related to an employee, the two result sets would be stitched together.

This type of query could be further complicated if optional cardinalities (outer joins) were specified. For example, you could have a left outer join on the sales fact from the order method dimension. This would return order methods that had sales as well as those that did not, which means it would be possible to see null values in both fact fields of the report.

Stitch Queries

- Stitch Query is an IBM Cognos term
- Uses conformed dimensions to "stitch" multi-fact queries together with a full outer join



Conformed dimensions contain the descriptive attributes and corresponding names, meanings, and values that have been agreed to across the enterprise.

Conformed dimensions can be used to merge two independent fact queries together, as seen in the slide example above, without loss of data. In this case, the Product dimension is used to stitch the results of query 1 and query 2 together on the common value selected from the Product dimension. Again, as seen on the previous page, no values are lost on either side of the query. Where the facts have the same data in common, both fact cells contain values. Where a fact does not share common data, the fact cells are null.

If the data source supports this type of query, the processing will be done at the data source level. If not, the processing will be done on the IBM Cognos servers.

We will examine the anatomy of a stitch query later in the course.

Business Analytics software

IBM

Cardinality in IBM Cognos (1 of 3)

- IBM Cognos uses cardinality:
 - for aggregation purposes
 - to identify which query subjects are dimensions and which are facts

```

graph TD
    EMPLOYEE[EMPLOYEE] -- "1..1" --> OH[ORDER_HEADER]
    OM[ORDER_METHOD] -- "1..1" --> OH
    OH -- "1..n" --> OD[ORDER_DETAILS]
    OH -- "1..n" --> FDQ["Fact or Dimension?  
Depends on the query"]
  
```

© 2015 IBM Corporation

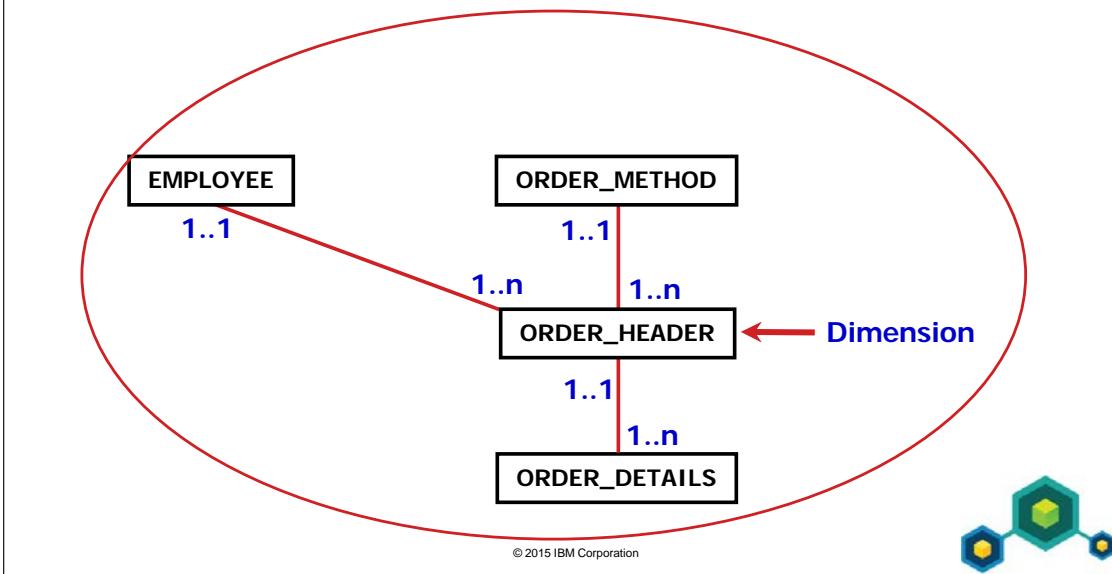
Typically, facts are found on the 1..n side of a query. The facts found in these tables can then be aggregated if requested by the user.

For every generated query, IBM Cognos identifies each query subject as either a "fact" or a "dimension" based on cardinality. Query subjects with only 0..n or 1..n cardinalities attached are identified as facts and query subjects with 0..1 or 1..1 cardinalities attached are identified as dimensions. Query subjects with both types of cardinality have the potential to be ambiguous. This is the case with ORDER_HEADER.

Note that, although query subjects may contain no measures, such as ORDER_HEADER, IBM Cognos BI will view them as fact tables based on cardinality. Treating the query subjects on the 1..n side on the cardinality as facts is a logical choice when trying to apply aggregation rules since measures are usually on this end of the cardinality. If they were on the 1..1 side, there would not be very much to aggregate.

Cardinality in IBM Cognos (2 of 3)

- Examine a query using these query subjects:



Again, query usage is applied based on which query subjects are used in a query.

In the slide example, **ORDER_HEADER** has both **1..1** and **1..n** cardinalities attached. Since fact query subjects must have only **1..n** or **0..n** cardinalities attached, **ORDER_HEADER** will be treated as a dimension in this scenario.

Business Analytics software

IBM

Cardinality in IBM Cognos (3 of 3)

- Examine a query using these query subjects

```

classDiagram
    class EMPLOYEE
    class ORDER_METHOD
    class ORDER_HEADER
    class ORDER_DETAILS

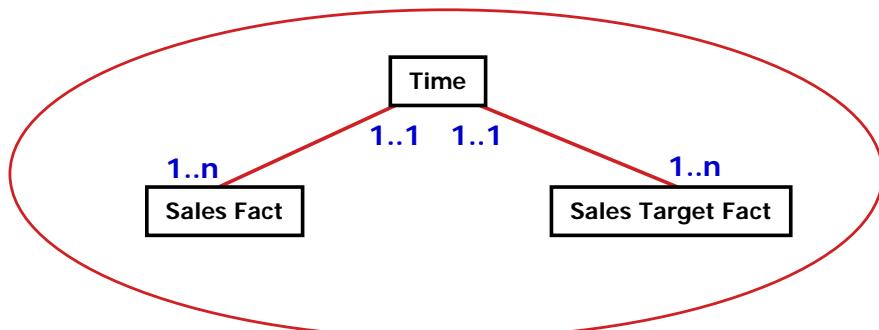
    EMPLOYEE "1..1" -- "1..n" ORDER_HEADER
    ORDER_METHOD "1..1" -- "1..n" ORDER_HEADER
    ORDER_HEADER "1..n" -- "1..n" ORDER_DETAILS
    
```

© 2015 IBM Corporation

In this query scenario, ORDER_HEADER has only 1..n cardinalities attached. Therefore it will be treated as a fact.

When Does IBM Cognos Generate Stitch Queries?

Time is a conformed dimension



Sales Fact and Sales Target Fact have only 1..n cardinality attached and are therefore treated as facts and stitched together in this query

© 2015 IBM Corporation



When one or more conformed dimensions are used in a multi-fact query (where query subjects are identified as facts by the IBM Cognos query engine), a stitch query (full outer join) will occur.

A stitch query is required when you have different levels of granularity between the facts and do not want to lose uncommon rows of data.

Business Analytics software

IBM

When You May Not Want a Stitch Query

When the level of granularity is the same for both facts, a stitch query (full outer join) may be unnecessary

```

graph TD
    Employee[Employee] ---|1..1| Pay[Pay]
    Employee[Employee] ---|1..1| FTD[Federal Tax Deduction]
    Pay[Pay] ---|1..1| FTD[Federal Tax Deduction]
  
```

Pay and Federal Tax Deduction will not be identified as facts due to cardinality and therefore no stitch query will be performed

© 2015 IBM Corporation

In cases where the facts are at the same level of granularity, you may want to avoid the extra processing of a full outer join and simply configure 1..1 to 1..1 relationships between your dimension and fact query subjects.

You must be extremely familiar with your data and the expected results when implementing this type of configuration. Use this approach with caution as there is potential to prevent required stitch queries as the model scales and involves dimensions with different levels of granularity.

Identify Reporting Traps

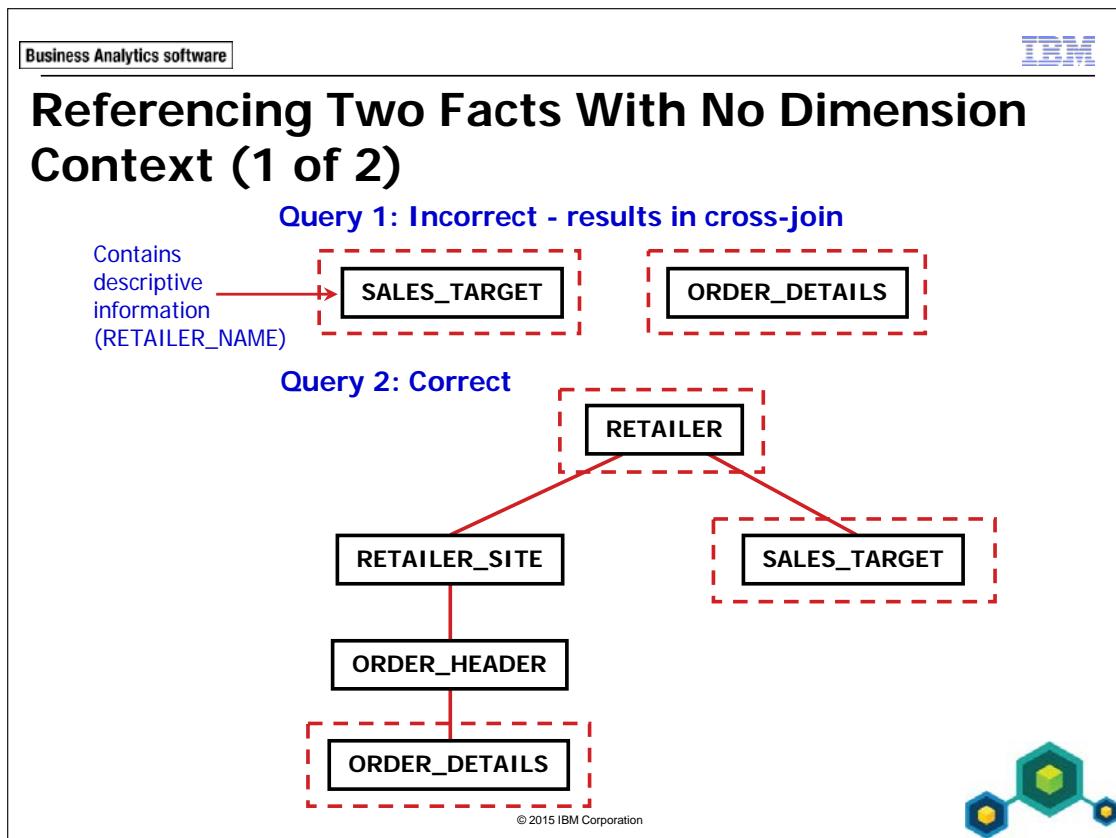
- Reporting traps can occur when you:
 - reference two facts with no dimension context
 - generate unwanted query splits (full outer joins)
 - write reports with query subjects that have ambiguous joins

© 2015 IBM Corporation



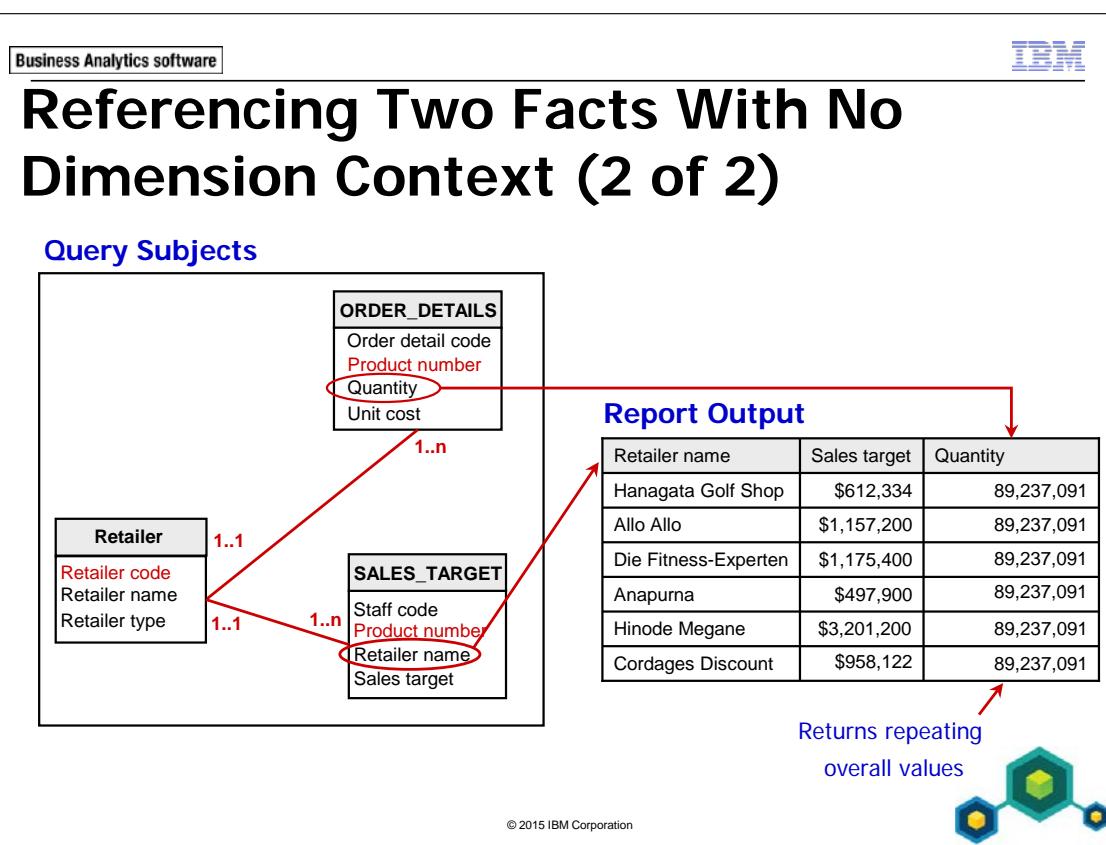
When beginning to model for predictable results, you will want to initially identify any reporting traps that exist within your model.

We will cover these reporting traps throughout this module.



A query on two facts with no dimension context is treated as a cross-product join. This brings duplicate data from the two datasets together in one query. Authors can accidentally author these types of queries if you have left textual (dimensional information) items in a fact query subject. For example, if Retailer name was a query item in **SALES_TARGET**, authors might choose this item thinking it also applies to **ORDER_DETAILS**.

This can be avoided if textual items are removed from the fact query subjects. Doing so will force authors to use at least one common dimension in the query to join the facts together. In Query 2, retailer information is taken from **RETAILER**.



The query above returns the distinct overall total of Quantity and applies it to each Retailer name row.

Again, in this scenario, authors can accidentally misuse Retailer name from the SALES_TARGET fact query subject and cause unexpected results. Taking Retailer name from the conformed dimension would generate a stitch query and provide expected results.

Demo 1: Reference Two Facts with No Dimension Context

Purpose:

Business Analysts often need to generate a report that compares sales targets against actual sales. You will test such a report in order to identify possible reporting traps.

Components: **Framework Manager, IBM Cognos Report Studio**

Project: **GO Operational**

Package: **GO Operational**

Task 1. Import SALES_TARGET into the gosales namespace.

1. In **Framework Manager**, open the **GO Operational** project located at **C:\Edcognos\B5A52\CBIFM-Start Files\Module 6\GO Operational**. Log in as User ID admin, Password Education1, if prompted.
2. In the **Project Viewer**, expand **GO Operational Model > Foundation Objects view**.
3. Right-click the **gosales** namespace, and then click **Run Metadata Wizard**. You will import more items from the GOSALES database.
4. Click **Next** twice.
5. Expand **GOSALES > Tables**, select **SALES_TARGET**, and then click **Next**.
6. On the **Generate Relationships** page, click the **Both** radio button, and then click **Import**.
7. Click **Finish**.

The gosales namespace now contains the SALES_TARGET query subject.

Task 2. Test a Targets vs Actual Revenue report.

1. Publish the **GO Operational** package.
2. In **IBM Cognos Connection**, launch **IBM Cognos Report Studio**, and use the **GO Operational** package to create a **List** report.

3. Expand **Foundation Objects View > gosales**, and add the following query items to the report:

Query Subject	Query Item
SALES_TARGET	RETAILER_NAME SALES_TARGET
ORDER_DETAILS	QUANTITY UNIT_SALE_PRICE

You could have taken COMPANY_NAME from RETAILER, but to identify potential areas of unpredictability for authors you will take it from SALES_TARGET.

4. Click the arrow beside the **Run** button and select **Run Report – HTML**.

The results appear as follows:

RETAILER_NAME	SALES_TARGET	QUANTITY	UNIT_SALE_PRICE
1 for 1 Sports shop	5,666,300	89,237,091	\$120.12
4 Golf only	4,385,300	89,237,091	\$120.12
4 Your Eyes	766,800	89,237,091	\$120.12
Aarhus Sport	5,288,300	89,237,091	\$120.12
Accapamento	5,338,900	89,237,091	\$120.12
Accesorios Importados, S.A. de C.V.	6,998,100	89,237,091	\$120.12
AcquaVerde	11,309,000	89,237,091	\$120.12
Act'N'Up Fitness	12,088,300	89,237,091	\$120.12
Actiforme	1,827,700	89,237,091	\$120.12
Action Factory	7,514,300	89,237,091	\$120.12

The resulting report has the same QUANTITY and UNIT_SALE_PRICE for every row, which is unexpected. To complete the report, you would create a calculation of Quantity multiplied by Unit Sale Price to get Actual Revenue values, but there is no need to continue since an issue has already been found.

5. Close the browsers without saving the report.

Task 3. Analyze the query in Framework Manager.

1. In **Framework Manager**, in the **Diagram**, locate **SALES_TARGET** and **ORDER_DETAILS**.

There are no direct relationships between these two query subjects. To see how they are joined in a query, you will recreate the same query in Framework Manager and view the generated SQL.

2. In the **Project Viewer**, select and **Test** the following query items with **Auto Sum** enabled:

Query Subject	Query Item
SALES_TARGET	RETAILER_NAME SALES_TARGET
ORDER_DETAILS	QUANTITY UNIT_SALE_PRICE

3. Click the **Query Information** tab.

The results appear as follows:

```
Cognos SQL
select
    D2.RETAILER_NAME as RETAILER_NAME,
    D2.SALES_TARGET as SALES_TARGET,
    D3.QUANTITY as QUANTITY,
    D3.UNIT_SALE_PRICE as UNIT_SALE_PRICE
from
    (select
        SALES_TARGET.RETAILER_NAME as RETAILER_NAME,
        XSUM(SALES_TARGET.SALES_TARGET) for SALES_TARGET.RETAILER_NAME
    from
        GOSALES..GOSALES.SALES_TARGET SALES_TARGET
    group by
        SALES_TARGET.RETAILER_NAME
    ) D2,
    (select distinct
        XSUM(ORDER_DETAILS.QUANTITY) as QUANTITY,
        XAVG(ORDER_DETAILS.UNIT_SALE_PRICE) as UNIT_SALE_PRICE
    from
        GOSALES..GOSALES.ORDER_DETAILS ORDER_DETAILS
    ) D3
```

The query has two sub queries (aliased as D2 and D3) that are not joined. The two queries are simply separated by a comma with no join conditions. This is known as a cross-product join. In this case, one query is on all sales targets grouped by retailer name, and the other is on a distinct value for all order quantities and unit sale prices with no grouping. This explains why all the quantities and prices are the same. IBM Cognos does not join the two query structures because they are both facts and there is no conformed (shared) dimension joining them.

You will need an alternate source for RETAILER_NAME, so you will try the only other instance of RETAILER_NAME in the project, the one in ORDER_HEADER.

4. Click **Close**.

Task 4. Attempt a resolution.

1. Select and **Test** the following query items with **Auto Sum** enabled:

Query Subject	Query Item
ORDER_HEADER	RETAILER_NAME
SALES_TARGET	SALES_TARGET
ORDER_DETAILS	QUANTITY UNIT_SALE_PRICE

The results appear as follows:

Test results			
RETAILER_NAME	SALES_TARGET	QUANTITY	UNIT_SALE_PRICE
1 for 1 Sports shop	4205368540	164233	99.01926928281461
4 Golf only	4205368540	53570	179.48930379746835
Aarhus Sport	4205368540	117117	114.0472600619195
Accapamento	4205368540	152551	104.36630165289256

You can now see separate quantities and unit sale prices for each retailer, but the sales targets are now all identical.

2. Click the **Query Information** tab.

The results appear as follows:

```
Cognos SQL
select
    D3.RETAILER_NAME as RETAILER_NAME,
    D2.SALES_TARGET as SALES_TARGET,
    D3.QUANTITY as QUANTITY,
    D3.UNIT_SALE_PRICE as UNIT_SALE_PRICE
from
    (select distinct
        XSUM(SALES_TARGET.SALES_TARGET) as SALES_TARGET
    from
        GOSALES..GOSALES.SALES_TARGET SALES_TARGET
    ) D2,
    (select
        ORDER_HEADER.RETAILER_NAME as RETAILER_NAME,
        XSUM(ORDER_DETAILS.QUANTITY for ORDER_HEADER.RETAILER_NAME)
        XAVG(ORDER_DETAILS.UNIT_SALE_PRICE for ORDER_HEADER.RETAILER_NAME)
    from
        GOSALES..GOSALES.ORDER_HEADER ORDER_HEADER,
        GOSALES..GOSALES.ORDER_DETAILS ORDER_DETAILS
    where
        (ORDER_HEADER.ORDER_NUMBER = ORDER_DETAILS.ORDER_NUMBER)
    group by
        ORDER_HEADER.RETAILER_NAME
    ) D3
```

Once again you have two subqueries that are not joined. However, this time it is sales targets that are not grouped by retailer name. IBM Cognos still does not join the two query subjects because you have not provided a conformed dimension.

If you speak with the database administrator, you will find out that COMPANY_NAME in the RETAILER query subject holds the retailer name that can be used between the two fact query subjects. You will try this in the next task.

3. Click **Close**.

Task 5. Correct the query.

- Select and **Test** the following query items with **Auto Sum** enabled:

Query Subject	Query Item
RETAILER	COMPANY_NAME
SALES_TARGET	SALES_TARGET
ORDER_DETAILS	QUANTITY UNIT_SALE_PRICE

The results appear as follows:

Test results			
COMPANY_NAME	SALES_TARGET	QUANTITY	UNIT_SALE_PRICE
1 for 1 Sports shop	5666300	164233	99.01926928281461
4 Golf only	4385300	53570	179.48930379746835
4 Your Eyes	766800	14928	106.64756756756756
Aarhus Sport	5288300	117117	114.0472600619195
Accapamento	5338900	152551	104.36630165289256

You can now see separate targets, quantities, and prices for each retailer. If you were to create this report in Query Studio, you would now calculate Actual Revenue to compare against the targets. (You will not do this now in order to continue investigating modeling issues and will create a Revenue calculation in the model later in the course.)

2. Click the **Query Information** tab.

The results appear as follows:

```
Cognos SQL
select
    coalesce(D2.COMPANY_NAME,D3.COMPANY_NAME) as COMPANY_NAME,
    D2.SALES_TARGET as SALES_TARGET,
    D3.QUANTITY as QUANTITY,
    D3.UNIT_SALE_PRICE as UNIT_SALE_PRICE
from
    (select
        RETAILER.COMPANY_NAME as COMPANY_NAME,
        XSUM(ORDER_DETAILS.QUANTITY for RETAILER.COMPANY_NAME ) as QUANTITY,
        XAVG(ORDER_DETAILS.UNIT_SALE_PRICE for RETAILER.COMPANY_NAME ) as UNIT_SALE_PRICE
    from
        GOSALES..GOSALESRT.RETAILER RETAILER,
        GOSALES..GOSALES.ORDER_DETAILS ORDER_DETAILS,
        GOSALES..GOSALESRT.RETAILER_SITE RETAILER_SITE,
        GOSALES..GOSALES.ORDER_HEADER ORDER_HEADER
    where
        (RETAILER.RETAILER_CODE = RETAILER_SITE.RETAILER_CODE) and
        (RETAILER_SITE.RETAILER_SITE_CODE = ORDER_HEADER.RETAILER_SITE_CODE) and
        (ORDER_HEADER.ORDER_NUMBER = ORDER_DETAILS.ORDER_NUMBER)
    group by
        RETAILER.COMPANY_NAME
    ) D3
    full outer join
    (select
        RETAILER.COMPANY_NAME as COMPANY_NAME,
        XSUM(SALES_TARGET.SALES_TARGET for RETAILER.COMPANY_NAME ) as SALES_TARGET
    from
        GOSALES..GOSALESRT.RETAILER RETAILER,
        GOSALES..GOSALES.SALES_TARGET SALES_TARGET
    where
        (RETAILER.RETAILER_CODE = SALES_TARGET.RETAILER_CODE)
    group by
        RETAILER.COMPANY_NAME
    ) D2
    on (D3.COMPANY_NAME = D2.COMPANY_NAME)
```

There are two sub queries joined by COMPANY_NAME (retailer name) through a full outer join. Again, in this case, a full outer join is expected since you are stitching two separate fact queries together.

The query scenarios you just went through support the concept of ensuring descriptive data is retrieved from dimensions and not facts.

3. Close the test results window, and then in the **Diagram**, review the join paths for this query.

RETAILER goes through RETAILER_SITE and ORDER_HEADER to get to ORDER_DETAILS, and has a direct relationship with SALES_TARGET.

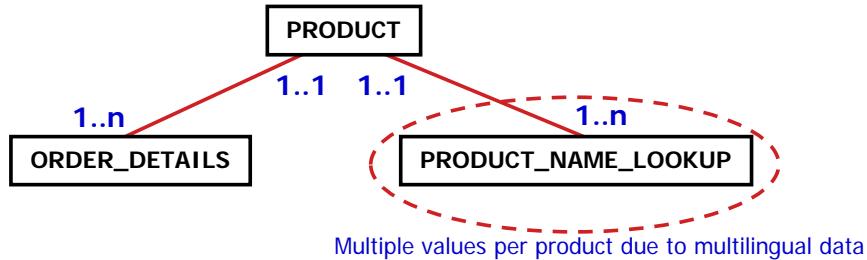
4. Leave Framework Manager open for the next demo.

Results:

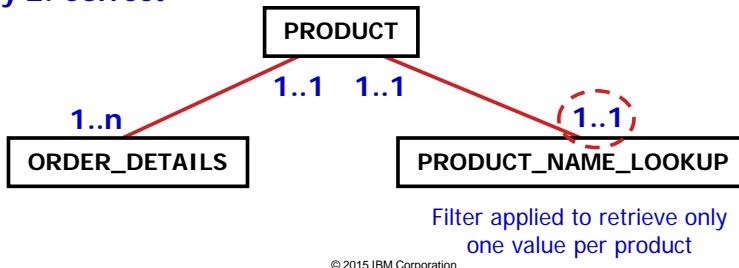
You tested a query to compare sales targets against actual sales, identified a reporting trap, and then identified a proper query to return expected results.

Generating Unwanted Query Splits

Query 1: Incorrect - a dimension is treated as a fact



Query 2: Correct



© 2015 IBM Corporation

Unwanted query splits (full outer joins) occur when query subjects (such as PRODUCT_NAME_LOOKUP in Query 1) are incorrectly identified as facts.

In this case PRODUCT_NAME_LOOKUP contains several values for each product in the PRODUCT table, one for each supported language in this multilingual table. The intent is for PRODUCT_NAME_LOOKUP to behave as a lookup table and return one value per product. Instead it acts as a fact based on cardinality.

In Query 2, the cardinality for PRODUCT_NAME_LOOKUP has been changed to 1..1 after a filter is put in place to ensure only one value is returned. In this scenario, it will always act as a dimension.

This issue will be resolved in the Create Calculations and Filters module.

Demo 2: Identify Unwanted Query Splits

Purpose:

**Authors will want to report on the overall quantity of products sold.
You will test such a query to identify a possible reporting trap.**

Component: **Framework Manager**

Project: **GO Operational**

Task 1. Create a query to view total quantity sold by product.

1. In **Framework Manager**, in **Foundation Objects View > gosales**, select and **Test** the following query items with **Auto Sum** enabled:

Query Subject	Query Item
PRODUCT_NAME_LOOKUP	PRODUCT_NAME
PRODUCT	PRODUCT_NUMBER
ORDER_DETAILS	QUANTITY

In order to return a larger result set, you will alter the options.

2. Click **Options** in the lower right corner, under **Number of results**, change the value from **25** to **250**, and then click **OK**.

Other options will be covered later in the course.

3. Click Test Sample.

The results are similar to the following:

Test results		
PRODUCT_NAME	PRODUCT_NUMBER	QUANTITY
Abah-Abah Husky	47110	262615
Abrisol 15	93110	1026459
Abrisol 30	94110	1955570
Abrisol en bâton	92110	796610
Abrisol universel	95110	991486
Acumulator Firefly	55110	1332666
Aigle	148110	550366
Aigle	148120	417397
Aigle	148130	356431
Akumulator do lampy wspinaczkowej Ognik	55110	1332666
Akumulátor Světluška	55110	1332666
Alicate Granito	60110	473686
Alicate Rad Granito	62110	546929
Alicates Granito	60110	473686
Alinare pentru mușcături de insecte	100110	178094
Alivio Aloe	99110	159547
Alivio Calamina	98110	117166
Alivio Picaduras de insectos	100110	178094

Notice the different language name but the same quantity value for the same product number.

PRODUCT_NAME_LOOKUP is a table in the database that contains multiple rows for each product, one for each supported language. Because there are multiple rows, the relationship is currently correct. 1..1 from PRODUCT to 1..n to PRODUCT_NAME_LOOKUP. It is not that this query is incorrect, but it may not be what the author expected. Likely you would want to see one product, in one language, with one summarized QUANTITY value.

You will look at the SQL to see what occurred.

- Click the **Query Information** tab.

The results appear as follows:

```
Cognos SQL
select
    D2.PRODUCT_NAME as PRODUCT_NAME,
    coalesce(D2.PRODUCT_NUMBER,D3.PRODUCT_NUMBER) as PRODUCT_NUMBER,
    D3.QUANTITY as QUANTITY
from
    (select distinct
        PRODUCT_NAME_LOOKUP.PRODUCT_NAME as PRODUCT_NAME,
        PRODUCT_NAME_LOOKUP.PRODUCT_NUMBER as PRODUCT_NUMBER
     from
        GOSALES..GOSALES.PRODUCT_NAME_LOOKUP PRODUCT_NAME_LOOKUP
    ) D2
    full outer join
    (select
        ORDER_DETAILS.PRODUCT_NUMBER as PRODUCT_NUMBER,
        XSUM(ORDER_DETAILS.QUANTITY for ORDER_DETAILS.PRODUCT_NUMBER )
     from
        GOSALES..GOSALES.ORDER_DETAILS ORDER_DETAILS
    group by
        ORDER_DETAILS.PRODUCT_NUMBER
    ) D3
    on (D2.PRODUCT_NUMBER = D3.PRODUCT_NUMBER)
```

Notice the two sub queries that are being joined with a full outer join. One retrieves all the multilingual product names for each product number, and the other, summarized order quantities for each product number. The two queries are then stitched together. This is an unwanted query split, since the intent in this query is to return one product name per product and show the quantities sold for that product.

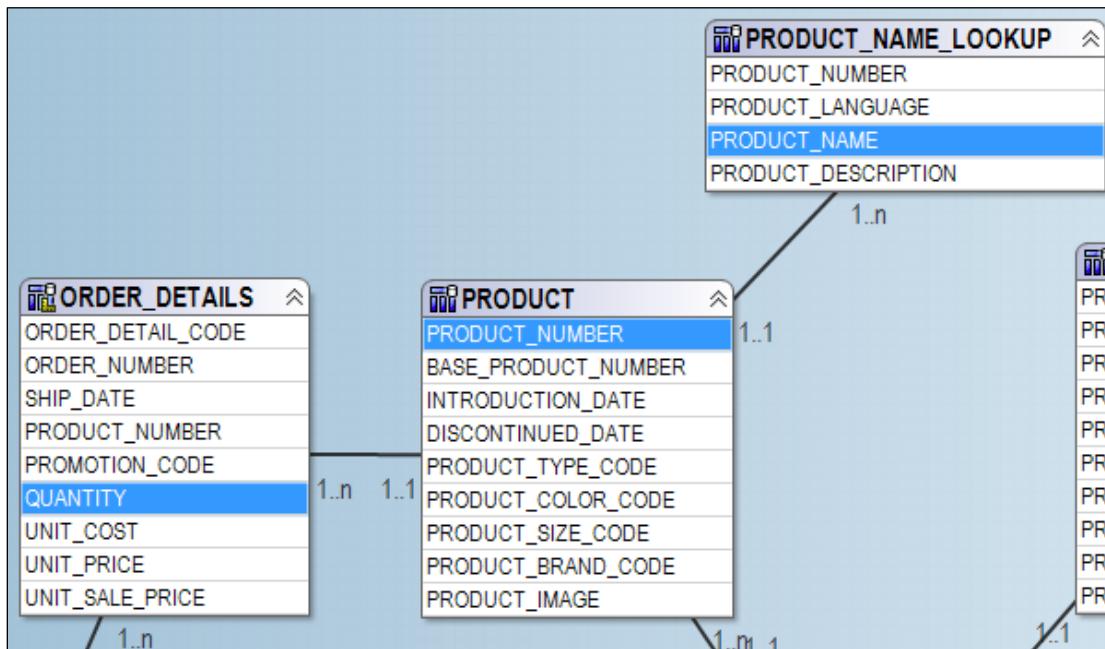
- Click **Close**.

Task 2. Examine the objects and their cardinality.

- In the middle pane, if necessary, click **Diagram**, and then double-click the **gosales** box to give it focus in the diagram.
You will increase the level of detail in the diagram to include query items.
- Click the **Auto Layout** icon, and set the **horizontal** and **vertical distance** to **25**.

3. In the diagram, locate **PRODUCT_NAME_LOOKUP**, **PRODUCT**, and **ORDER_DETAILS**.

The results appear similar to as follows:



In this scenario, **PRODUCT** is treated as a dimension (only 1..1 cardinalities attached), and **ORDER_DETAILS** and **PRODUCT_NAME_LOOKUP** are treated as facts (only 1..n cardinalities attached).

You will resolve the **PRODUCT_NAME_LOOKUP** issue later in the course by applying a filter and changing the cardinality.

4. Leave Framework Manager open for the next demo.

Results:

You tested a query for total Quantity sold by Product to identify a possible reporting trap, and found an unwanted query split.

Business Analytics software

IBM

Ambiguous Joins

- Query 1: Incorrect - a COUNTRY and ORDER_DETAILS query could use left or right path
- Query 2: Correct - a COUNTRY and ORDER_DETAILS query must specify left or right path

The diagram illustrates two scenarios regarding ambiguous joins. In the first scenario (Query 1), there are three entities: RETAILER_SITE, BRANCH, and COUNTRY. RETAILER_SITE and BRANCH are connected to ORDER_HEADER, which in turn connects to ORDER_DETAILS. There are two dashed red lines connecting RETAILER_SITE and BRANCH to COUNTRY, each ending in a question mark, indicating that either path could be taken. In the second scenario (Query 2), the COUNTRY entity is split into two distinct entities: RETAILER_COUNTRY and BRANCH_COUNTRY. RETAILER_SITE connects to both RETAILER_COUNTRY and BRANCH_COUNTRY. Both RETAILER_COUNTRY and BRANCH_COUNTRY connect to ORDER_HEADER, which then connects to ORDER_DETAILS. Dashed red lines connect RETAILER_SITE to both RETAILER_COUNTRY and BRANCH_COUNTRY, but no lines connect them directly to ORDER_DETAILS, thus avoiding ambiguity.

© 2015 IBM Corporation

Remove the risk of taking the wrong query path by removing the ambiguity: Creating aliases of COUNTRY with the appropriate relationships allows report authors to choose the desired path, in this case order details by retailer country or branch country.

The GO Operational model would actually use four versions of COUNTRY (and of SALES_REGION above it). The third version is for the hierarchy of SALES_REGION to COUNTRY to BRANCH to EMPLOYEE_HISTORY to EMPLOYEE. And the fourth version is for the hierarchy of SALES_REGION to COUNTRY to SALES_TARGET. We have just shown two of the four hierarchies in the slide to keep the diagram simple.

Demo 3: Identify Ambiguous Joins

Purpose:

Another common requirement for the business analysts will be to report on sales figures by retailer country. You will test such a query to identify a possible reporting trap.

Component: **Framework Manager**

Project: **GO Operational**

Task 1. Create the report.

1. In **Foundation Objects View > gosales**, select and **Test** the following query items with **Auto Sum** enabled:

Query Subject	Query Item
COUNTRY	COUNTRY_EN
ORDER_DETAILS	QUANTITY

The results appear as follows:

COUNTRY_EN	QUANTITY
Australia	1599920
Austria	1660769
Belgium	1302716
Brazil	1741344
Canada	4052045
China	4413127
Finland	2784969
France	3948439
Germany	3662772
Italy	2617761
Japan	4644148
Korea	2250400

The goal is to create a query for the number of units sold in the country of each retailer. These results seem like they might be correct. But to be sure, you will look at the SQL generated for this query.

2. View the SQL on the **Query Information** tab.

The results appear as follows:

```
Cognos SQL
select
    COUNTRY.COUNTRY_EN as COUNTRY_EN,
    XSUM(ORDER_DETAILS.QUANTITY for COUNTRY.COUNTRY_EN ) as QUANTITY
from
    GOSALES..GOSALES.COUNTRY COUNTRY,
    GOSALES..GOSALES.ORDER_DETAILS ORDER_DETAILS,
    GOSALES..GOSALES.ORDER_HEADER ORDER_HEADER,
    GOSALES..GOSALES.BRANCH BRANCH
where
    (COUNTRY.COUNTRY_CODE = BRANCH.COUNTRY_CODE) and
    (BRANCH.BRANCH_CODE = ORDER_HEADER.SALES_BRANCH_CODE) and
    (ORDER_HEADER.ORDER_NUMBER = ORDER_DETAILS.ORDER_NUMBER)
group by
    COUNTRY.COUNTRY_EN
```

The query did not take the path you wanted (through RETAILER_SITE). Instead, it took the path through BRANCH to ORDER_HEADER to ORDER_DETAILS. So the query set returned the number of units sold in each sales branch country.

3. Click **Close**.

Task 2. Examine the objects and their relationships.

You will use a feature called Context Explorer that helps you isolate modeling objects for closer examination and testing. It is a focused subset of the overall diagram. You will first change the default behavior of the Context Explorer to only show objects you select and not all related objects of the items you select.

1. From the **Diagram** menu, click **Diagram settings**.
2. Click the **Context Diagram** tab, select **Show only the selected objects**, and then click **OK**.
3. In the **Project Viewer**, select the following query subjects:

COUNTRY

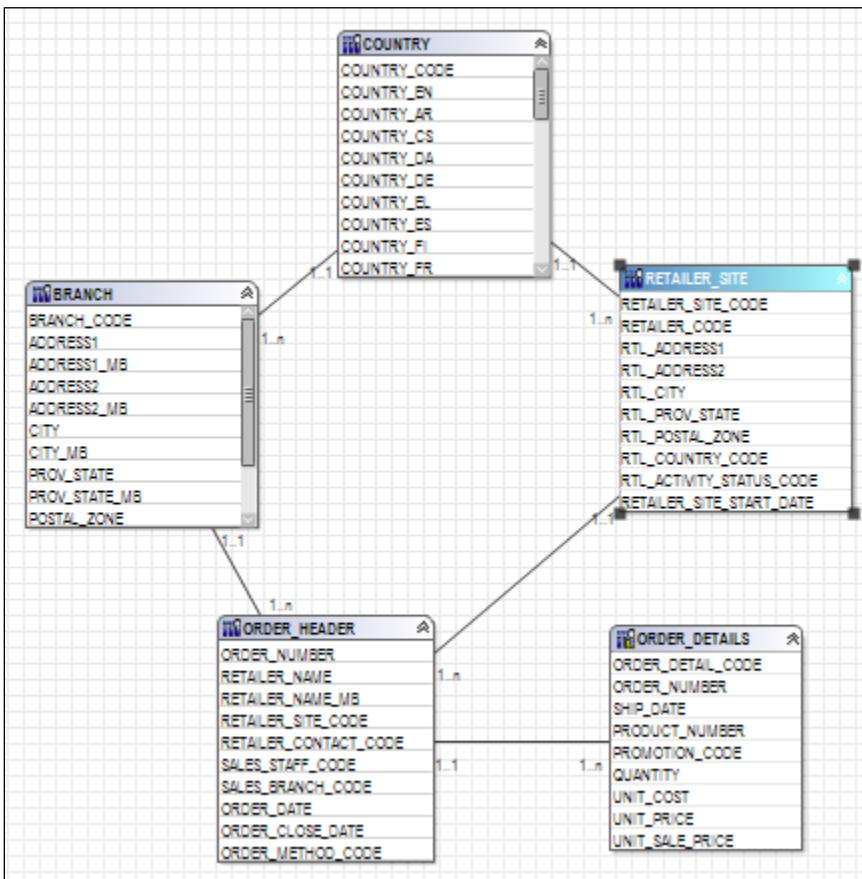
BRANCH

RETAILER_SITE

ORDER_HEADER

ORDER_DETAILS

4. Right-click one of the selected objects, and then click **Launch Context Explorer**.
5. Arrange the diagram as shown below:



When a query can take more than one path of equal length and all query subjects are in a single namespace or folder, IBM Cognos chooses the relationship that comes first alphabetically. Using modeling techniques such as alias query subjects, you can avoid all ambiguity.

You will apply a solution in the next module.

6. Close the **Context Explorer**, and leave Framework Manager open for the next demo.

Results:

You tested a query on sales by country, intending the countries to be for retailers. Because there were multiple query paths that could be used, you discovered the potential for unpredictable results.

Using Tools to Analyze the Model

- Verify Model: identifies errors
- Model Advisor: identifies areas needing review
- Model in freehand to identify:
 - which query subjects can act as either a fact or a dimension
 - fact query subjects that include descriptive data
 - unclear query paths
 - groupings of data (which items will be used to create a fact or dimension query subject for presentation to end users?)

© 2015 IBM Corporation



- Verify Model: After you select the conditions and severity for which you want to test, Verify Model identifies the errors and offers automated or manual repairs.
- Model Advisor: After you select the conditions for which you want to test, Model Advisor gives a clear, graphical presentation that summarizes the design issues uncovered and offers links to documentation for evaluating and resolving the issues.
- Modeling in freehand: Use a diagram printout to identify areas for concern and how you will build your final model. Concentrate on modeling dimensions first, since they are usually easy to identify, more problematic, and can resolve most of the modeling design issues.

When modeling in freehand, it is important to have an understanding of the business processes that created the facts. You cannot create a correct model without interaction with functional users and data modelers. Your requirements and understanding of the data should be clear.

Demo 4: Use Tools to Analyze the Model

Purpose:

Prior to modeling the metadata, you will analyze the model for errors and design issues.

Component: **Framework Manager**

Project: **GO Operational**

Task 1. Analyze for errors.

1. In the **Project Viewer**, right-click the **gosales** namespace, and then click **Verify Selected Objects**.

The Options tab lets you control two aspects of the verification process: the severity and the categories of verification. You will accept the defaults.

2. Click **Verify Model**.

No issues have been detected in this simple model. Note the Repair Selected tool, and the ability to group reported issues by object id, severity, or message description.

3. Click **Close**.

You can perform this verification process on any level under the root namespace of the project or verify the entire model from the Project menu.

Verification is also performed by default when publishing a package. All contents of the package are verified unless you deselect the option in the publish wizard.

Task 2. Analyze for design issues.

1. In **Project Viewer**, right-click **gosales** and then click **Run Model Advisor**.

The Options tab lets you control three forms of analysis: relationships, determinants, and miscellaneous checks. You will accept the defaults.

2. Click **Analyze**.

The Model Advisor identifies 3 issues:

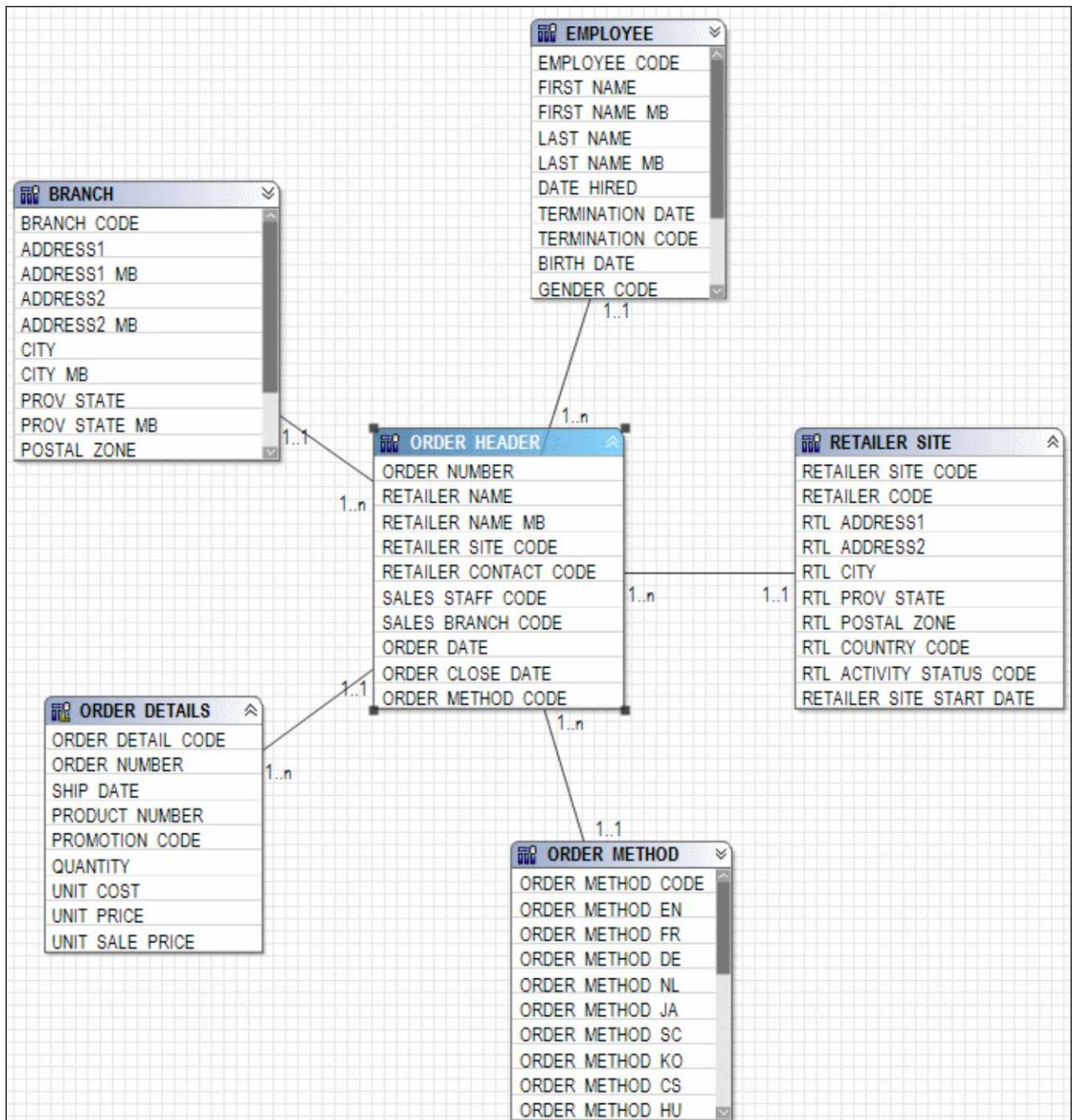
Issue 1 is "Facts identified by cardinality." You should confirm that the query subjects identified here are indeed facts. Both ORDER_DETAILS and SALES_TARGET are. But PRODUCT_NAME_LOOKUP is not a fact as defined by the data. EMPLOYEE_HISTORY is a factless fact, but is this how you want to use it for the reporting requirements, or should it be used as a lookup table for EMPLOYEE to return an employee's current job status? You'll look at these query subjects in more detail later in the course.

Issue 2 is "Query subjects that can behave as facts or dimensions." These items are on the 1..1 side of at least one relationship as well as being on the 1..n side of at least one relationship. These are also known as ambiguous query subjects. Before placing these in a package for reporting, you should investigate these objects to ensure that they do not have the potential of being used in an unpredictable manner. You have already seen that ORDER_HEADER may be used as a dimension in one query scenario and as a fact in another.

3. Under **Issue 2**, to the right of **ORDER_HEADER**, click **Launch Context**

Explorer 

4. If necessary, to view all query subjects, maximize the window and click **Fit All**.
 The results appear similar as follows:



Here you can quickly see how ORDER_HEADER relates to other query subjects and where potential problems might occur.

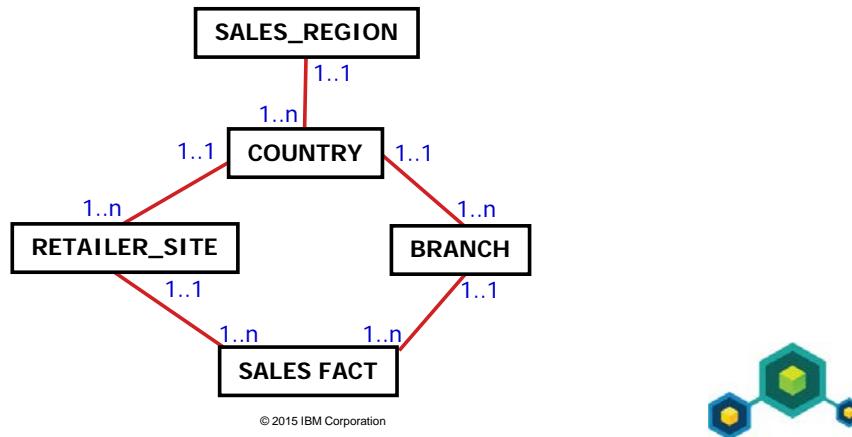
5. Close the **Context Explorer** and continue with the Model Advisor issues. Issue 3 is "Query subjects with multiple relationships." This section identifies query subjects that may cause ambiguous query paths. Ambiguity will be removed throughout the modeling process.
6. Close the **Model Advisor** dialog box, and leave Framework Manager open for the workshop.

Results:

Prior to modeling the metadata, you analyzed the model for errors and design issues.

Ambiguous Query Subjects (1 of 3)

- COUNTRY, RETAILER SITE, and BRANCH all appear ambiguous based on cardinality
- But these are logical hierarchies



It is important to understand when query subjects are really ambiguous and how to solve them.

There are logical hierarchies from SALES_REGION all the way to SALES FACT, but there is an ambiguous query path.

In this scenario, multiple cardinality types attached to the query subjects is not an issue, but a clear query path is.

Business Analytics software

IBM

Ambiguous Query Subjects (2 of 3)

- IBM Cognos understands hierarchy paths
- In this scenario, multiple cardinality types attached do not present a problem because there is a clear path that terminates at a fact

```

graph TD
    RR[RETAILER REGION] -- "1..1" --> RC[RETAILER COUNTRY]
    RR -- "1..n" --> RS[RETAILER SITE]
    RC -- "1..1" --> RS
    RC -- "1..n" --> SF[SALES FACT]
    BR[BRANCH REGION] -- "1..1" --> BC[BRANCH COUNTRY]
    BR -- "1..n" --> B[BRANCH]
    BC -- "1..1" --> B
    BC -- "1..n" --> SF
    SF --- C[© 2015 IBM Corporation]
    
```

Aliases

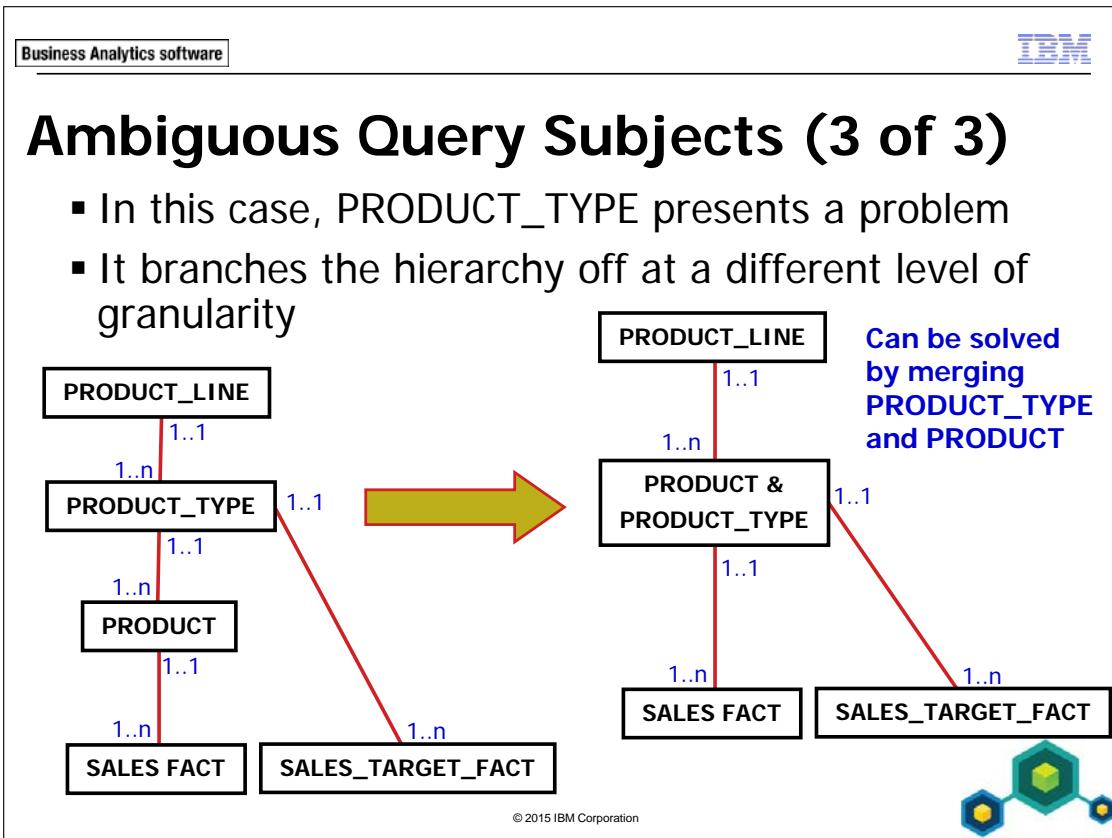
Aliases

© 2015 IBM Corporation

Using aliases you can remove the ambiguous query paths.

But the query subjects still have multiple cardinalities attached. This is OK because IBM Cognos can traverse the hierarchies and terminate at the actual fact, in this case SALES FACT.

This only becomes an issue if the hierarchy can branch off to different facts at different levels. You will see this example on the next page.



Remember querying PRODUCT_TYPE, PRODUCT, and SALES_TARGET_FACT caused PRODUCT to be treated as a fact query subject and generated an unwanted query split resulting in a full outer join? This is because IBM Cognos determined PRODUCT and SALES_TARGET_FACT were fact termination points and that PRODUCT_TYPE was a shared dimension between the two. Based on cardinality, this is the logical choice. But is it what you want?

After merging PRODUCT and PRODUCT_TYPE, there is a clear hierarchy path that terminates at the real fact query subjects, in this case, SALES FACT and SALES_TARGET_FACT.

Business Analytics software

IBM

Summary

- You should now be able to:
 - identify reporting traps
 - describe how IBM Cognos uses cardinality
 - identify when full outer joins are appropriate
 - use tools to analyze the model

© 2015 IBM Corporation

Workshop 1: Model in Freehand to Identify Query Usage

This workshop is based on modeling Recommendation 4: Model in freehand to identify query usage. This is an extremely important workshop as it lays the foundation of what will be modeled throughout this course.

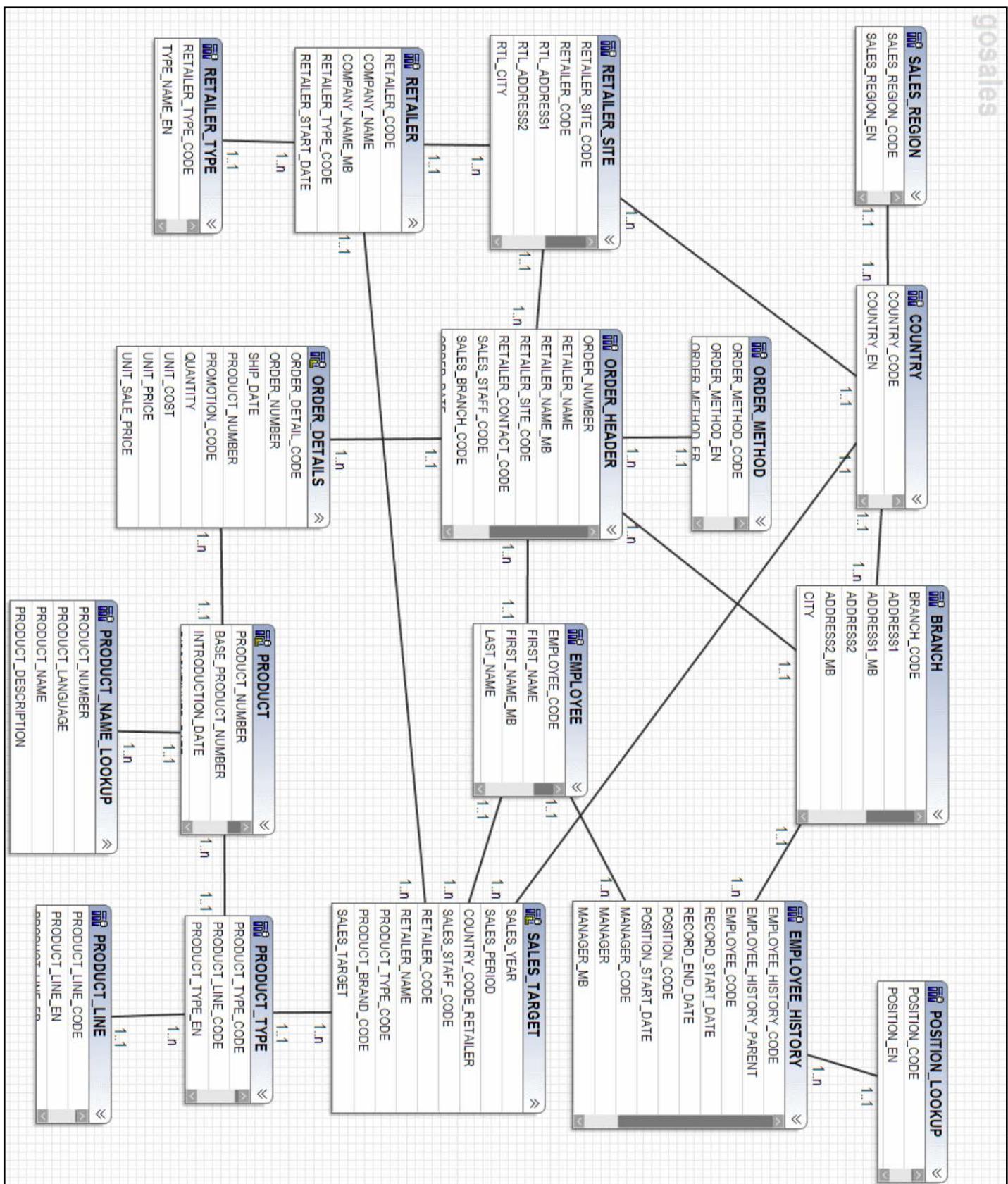
You have just been asked to analyze a Framework Manager model based on the Sample Outdoors Company operational database, and to make recommendations on how to enhance the model so that authors will achieve predictable results.

To do so, analyze the following three diagrams to identify:

- All groupings of related query structures (for example, PRODUCT_LINE, PRODUCT_TYPE, and PRODUCT). These can be consolidated into a single author-friendly model query subject for presentation
- All facts (query subjects whose relationships are all on the 1..n side)
- Any ambiguous query subjects that can behave as either facts or dimensions (these can be resolved through a process of creating aliases to remove ambiguous query paths or by merging query subjects).

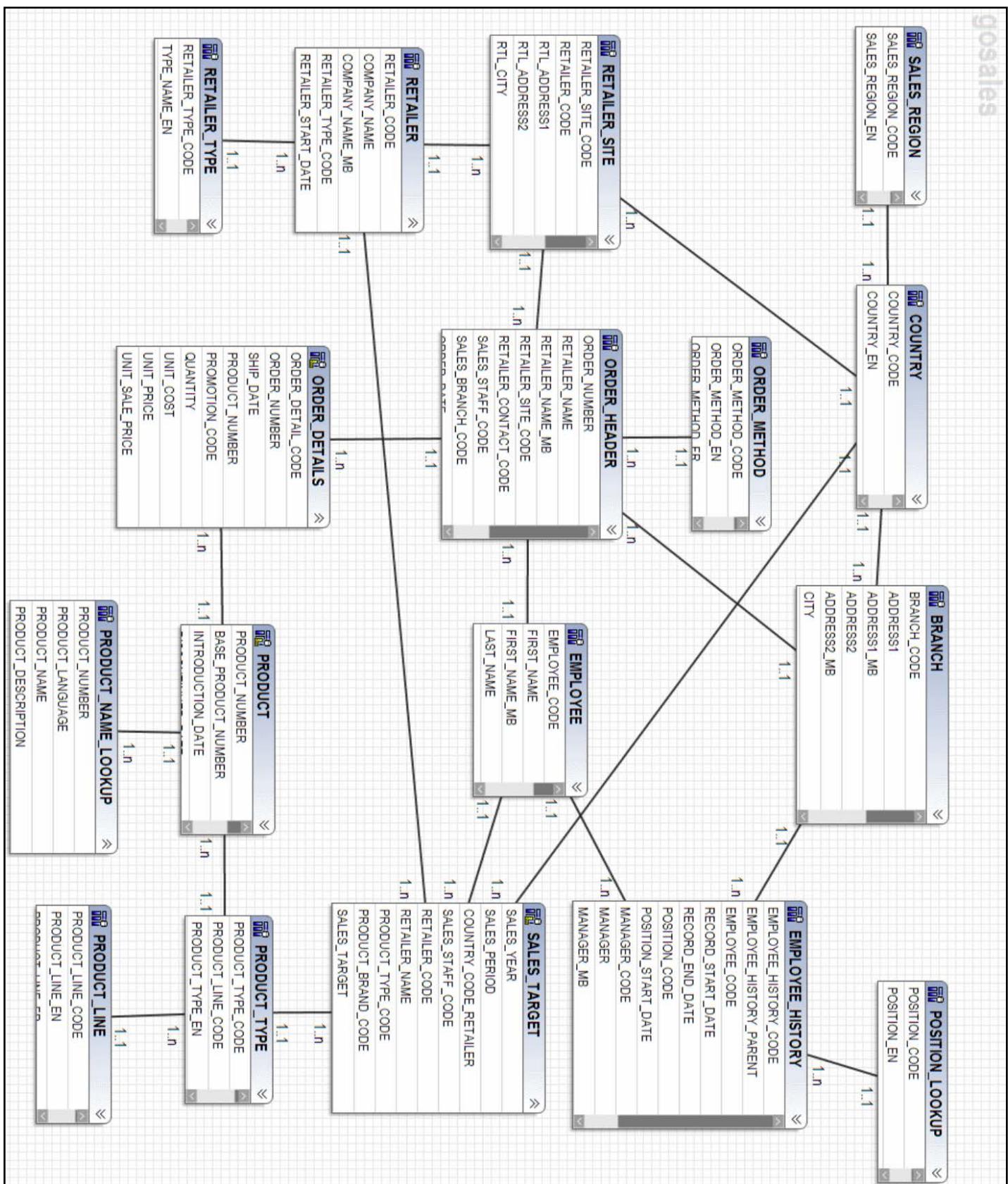
When done, close Framework Manager without saving the project.

Workshop 1: Identify Groupings



This material is meant for IBM Academic Initiative use only. NOT FOR RESALE

Workshop 1: Identify Facts

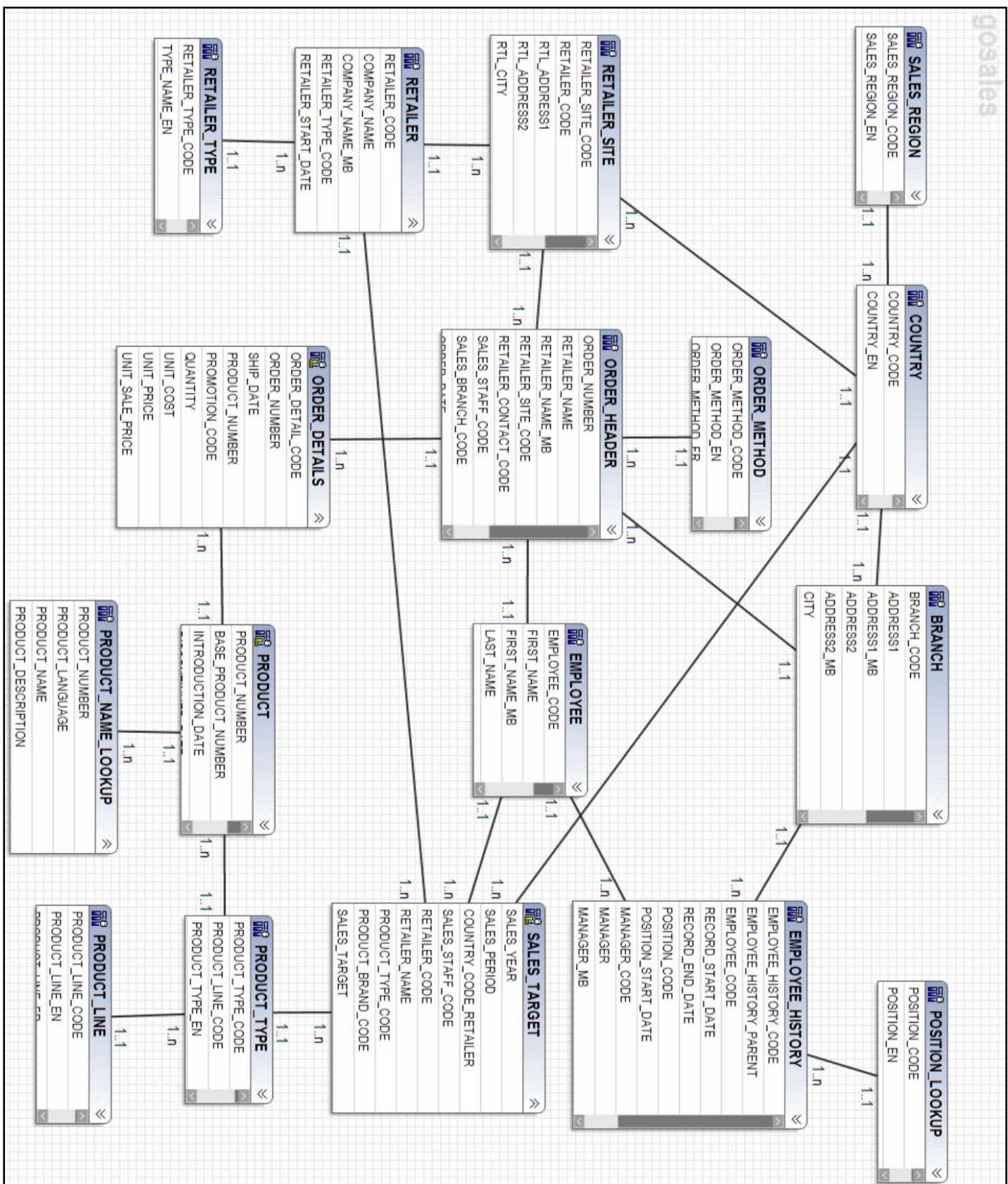


This material is meant for IBM Academic Initiative use only. NOT FOR RESALE

© 2003, 2015, IBM Corporation

This guide contains proprietary information which is protected by copyright. No part of this document may be photocopied, reproduced, or translated into another language without a legal license agreement from IBM Corporation.

Workshop 1: Identify Ambiguous Query Subjects



This material is meant for IBM Academic Initiative use only. NOT FOR RESALE

Workshop 1: Solution

These solutions will be explained in further detail and implemented throughout the next few modules.

Groupings:

The Identify Groupings page illustrates the first step in a star schema modeling solution, logically grouping related query structures:

- SALES_REGION, COUNTRY, BRANCH, EMPLOYEE_HISTORY, EMPLOYEE, and POSITION_LOOKUP to make up a **Staff by Location** dimension
- SALES_REGION, COUNTRY, RETAILER_SITE, RETAILER, and RETAILER_TYPE to make a **Retailer** dimension
- PRODUCT, PRODUCT_TYPE, PRODUCT_LINE and PRODUCT_NAME_LOOKUP to create a **Product** dimension

ORDER_METHOD is a stand-alone dimension and presents no issues.

There are also two other groupings, although not in the initial requirements, that may make sense. Sometimes during the modeling process the data reveals other alternatives that you can choose to take advantage. The ones found here are:

- SALES_REGION, COUNTRY and BRANCH to create a **Branch by Location** Dimension, which can then link to the grouping of ORDER_HEADER and ORDER_DETAILS in order to identify which branch a sale was made from
- SALES_REGION and COUNTRY to create a **Sales Target by Location** dimension

All of the above groupings will be used as dimensions to meet reporting requirements.

Facts:

ORDER_DETAILS is a fact as it only has 1..n cardinalities attached.

SALES_TARGET is a fact as it only has 1..n cardinalities attached.

EMPLOYEE_HISTORY is also a fact even though it contains no measures. However the requirements for this application are to report on employees' current positions. Later in the modeling process, a filter will be used to filter on the current position and therefore change the nature of the EMPLOYEE to EMPLOYEE_HISTORY relationship to 1..1 to 1..1.

PRODUCT_NAME_LOOKUP is identified as a fact as it only has 1..n cardinalities attached. In the data, this is not a fact and will be corrected with a filter that will change the cardinality to 1..1.

Ambiguous Query Subjects:

COUNTRY is used in multiple query paths and is a good candidate for aliasing.

BRANCH is used in two query paths and is a good candidate for aliasing.

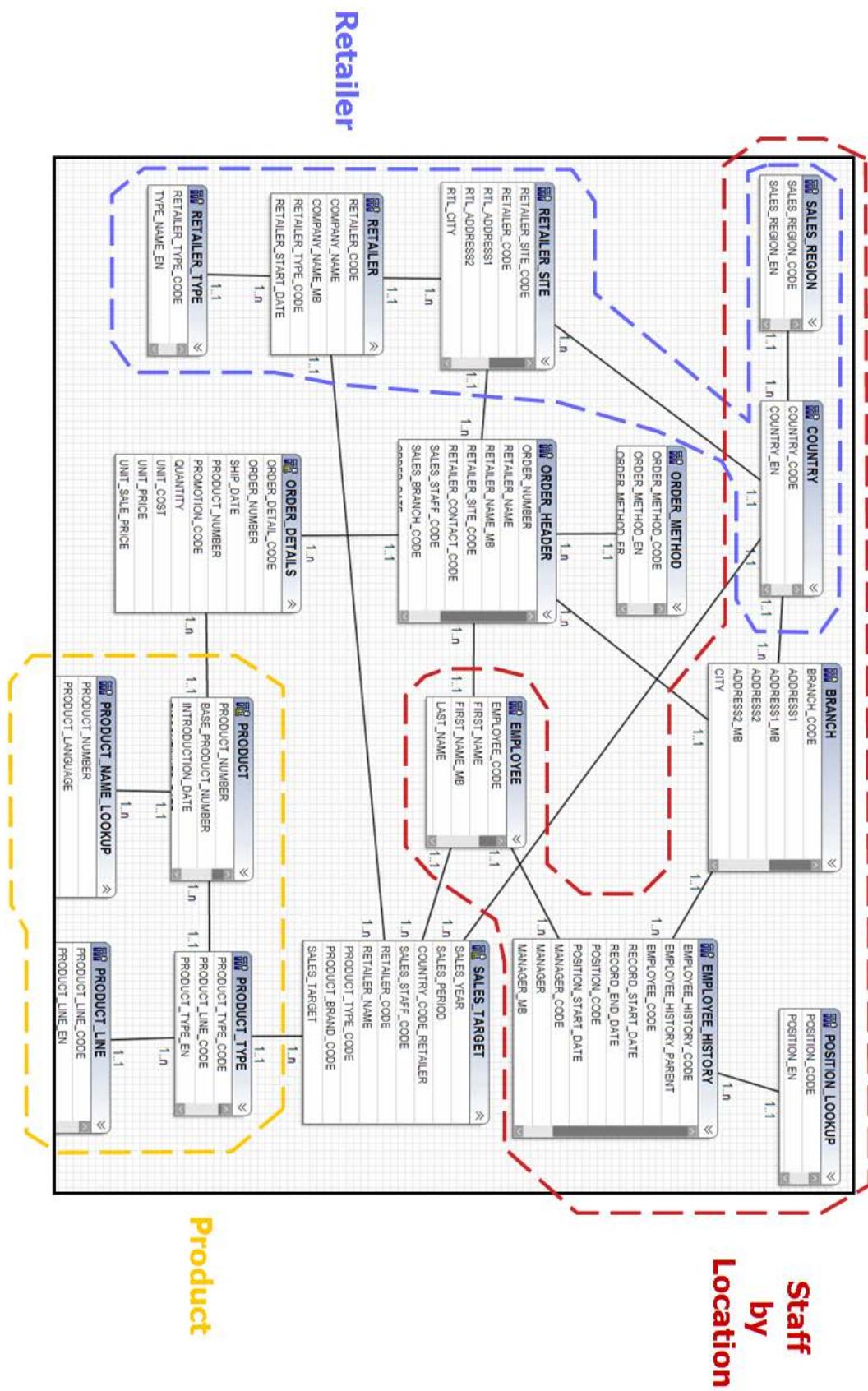
ORDER_HEADER is an ambiguous query subject because it has multiple cardinality types attached with multiple paths. It is not part of a hierarchy with one path.

- ORDER_HEADER is a query subject used as a bridge between most dimensions and the sales facts in ORDER_DETAILS. It contains keys allowing other dimensions to query order details.
- ORDER_HEADER is a good candidate for merging with ORDER_DETAILS to create a new fact query subject. Once these two query subjects are merged, the new query subject will have only 1..n cardinalities attached ensuring it will always be treated as a fact.

RETAILER has multiple cardinality types attached and branches the hierarchy off at a higher level of granularity. It is a good candidate for merging with RETAILER_SITE.

PRODUCT_TYPE has multiple cardinalities attached and branches off at a higher level of granularity. It is also a good candidate for merging with PRODUCT.

Workshop 1: Main Groupings

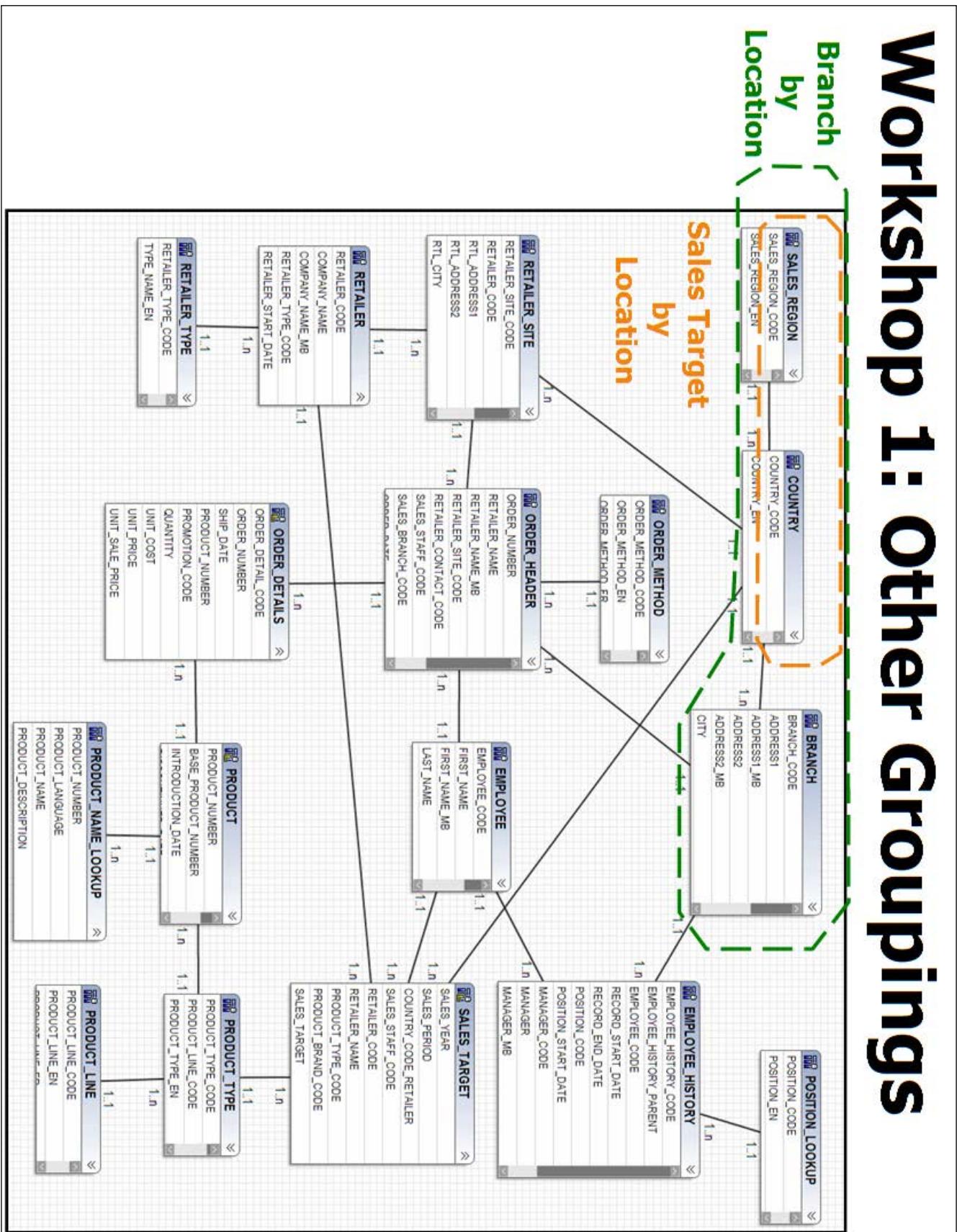


This material is meant for IBM Academic Initiative use only. NOT FOR RESALE

© 2003, 2015, IBM Corporation

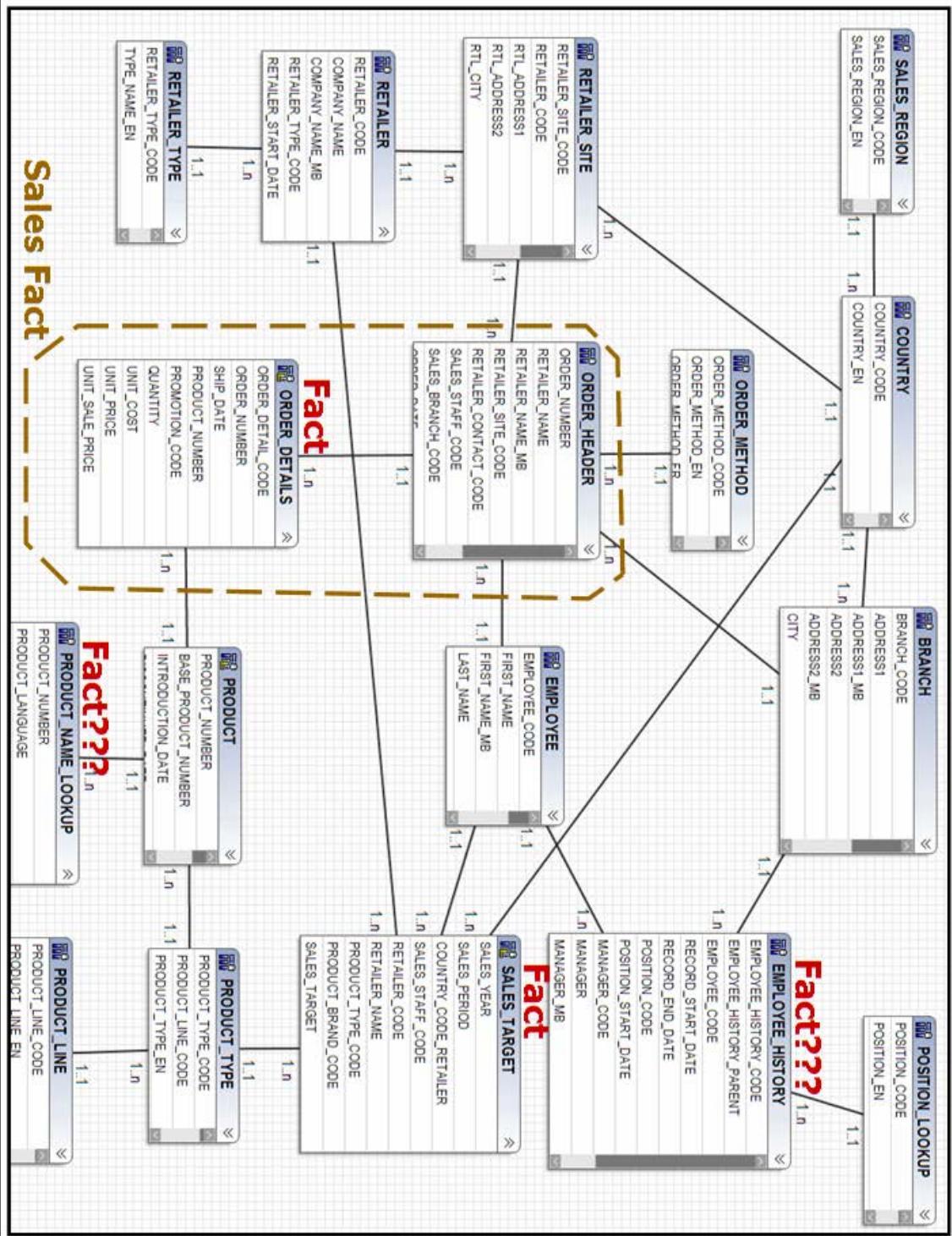
This guide contains proprietary information which is protected by copyright. No part of this document may be photocopied, reproduced, or translated into another language without a legal license agreement from IBM Corporation.

Workshop 1: Other Groupings



This material is meant for IBM Academic Initiative use only. NOT FOR RESALE

Workshop 1: Facts



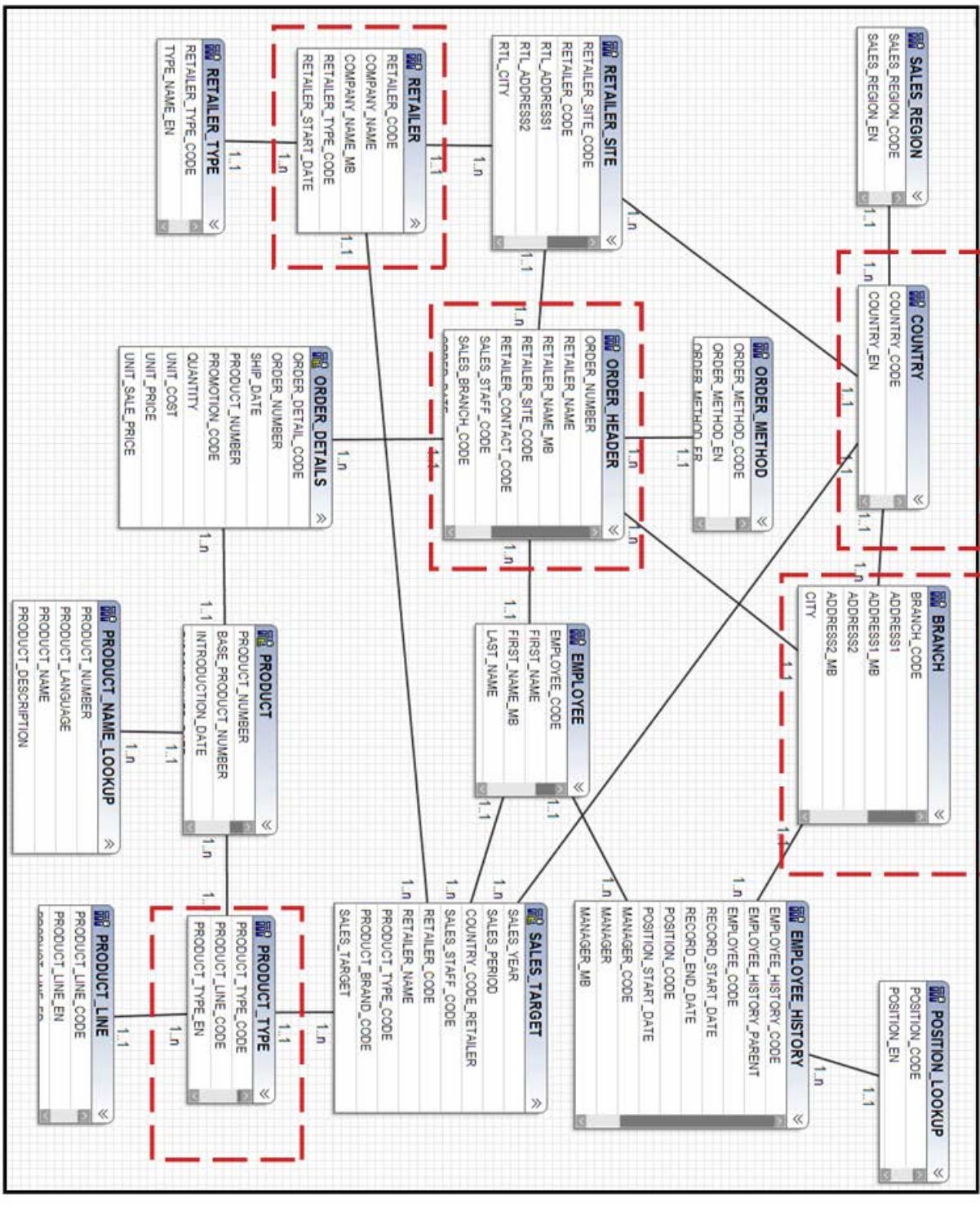
This material is meant for IBM Academic Initiative use only. NOT FOR RESALE

© 2003, 2015, IBM Corporation

This guide contains proprietary information which is protected by copyright. No part of this document may be photocopied, reproduced, or translated into another language without a legal license agreement from IBM Corporation.

6-51

Workshop 1: Ambiguous Query Subjects



This material is meant for IBM Academic Initiative use only. NOT FOR RESALE



Modeling for Predictable Results: Virtual Star Schemas

IBM Cognos BI



Business Analytics software

© 2015 IBM Corporation

This material is meant for IBM Academic Initiative use only. NOT FOR RESALE

© 2003, 2015, IBM Corporation

This guide contains proprietary information which is protected by copyright. No part of this document may be photocopied, reproduced, or translated into another language without a legal license agreement from IBM Corporation.

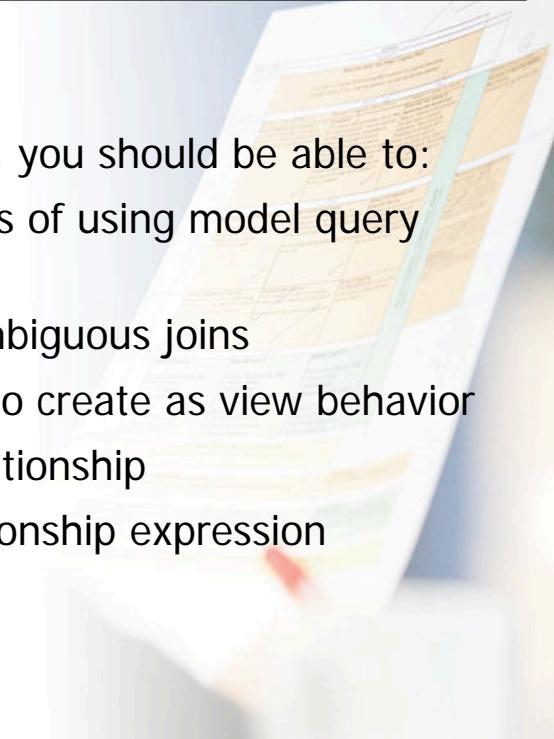
Business Analytics software

IBM

Objectives

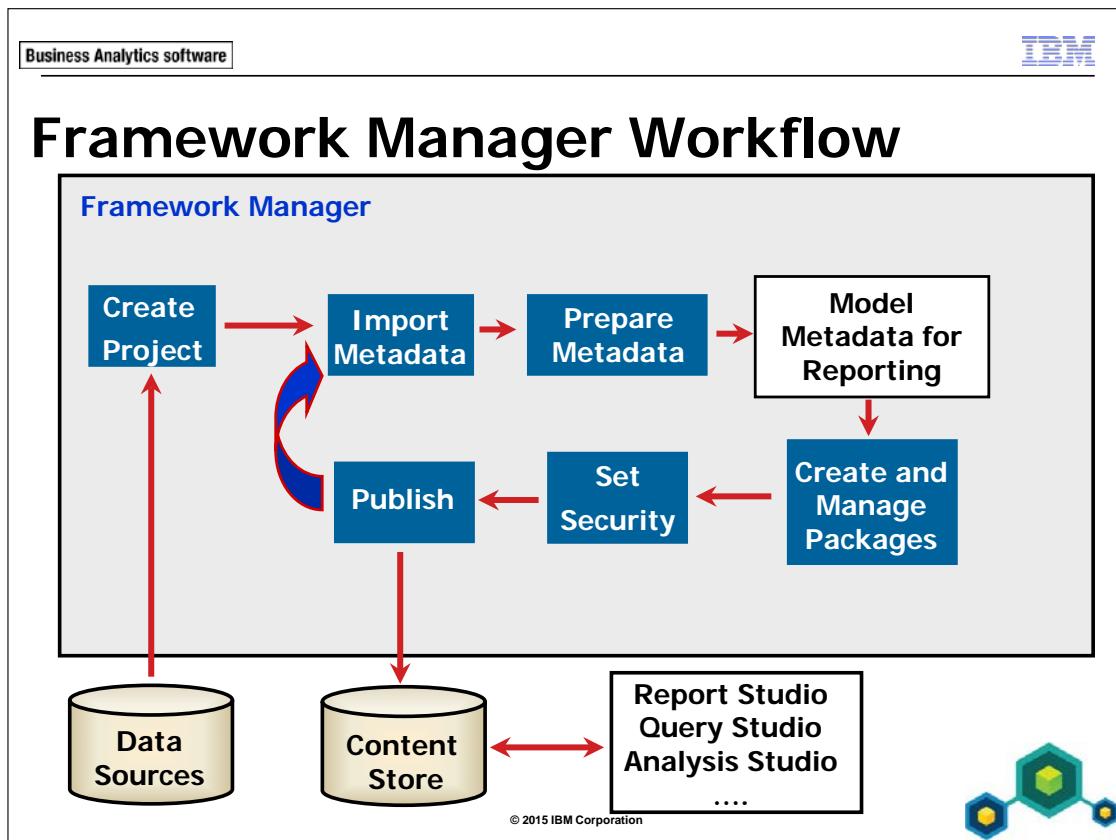
- At the end of this module, you should be able to:
 - understand the benefits of using model query subjects
 - use aliases to avoid ambiguous joins
 - merge query subjects to create as view behavior
 - resolve a recursive relationship
 - create a complex relationship expression

© 2015 IBM Corporation



Unless otherwise specified in demo or workshop steps, you will always log on to IBM Cognos BI in the Local LDAP namespace using the following credentials:

- User ID: admin
- Password: Education1



This module discusses modeling metadata as a virtual star schema in order to ensure predictable results.

Modifying Data Source Query Subjects

- to modify a data source query subject, create a model query subject that is based on it
- can make the SQL more "complex" and cause IBM Cognos to request metadata from the database
- when you add relationships to model query subjects, encapsulated joins are honored
- after modifying query subjects, view the generated SQL to verify the results

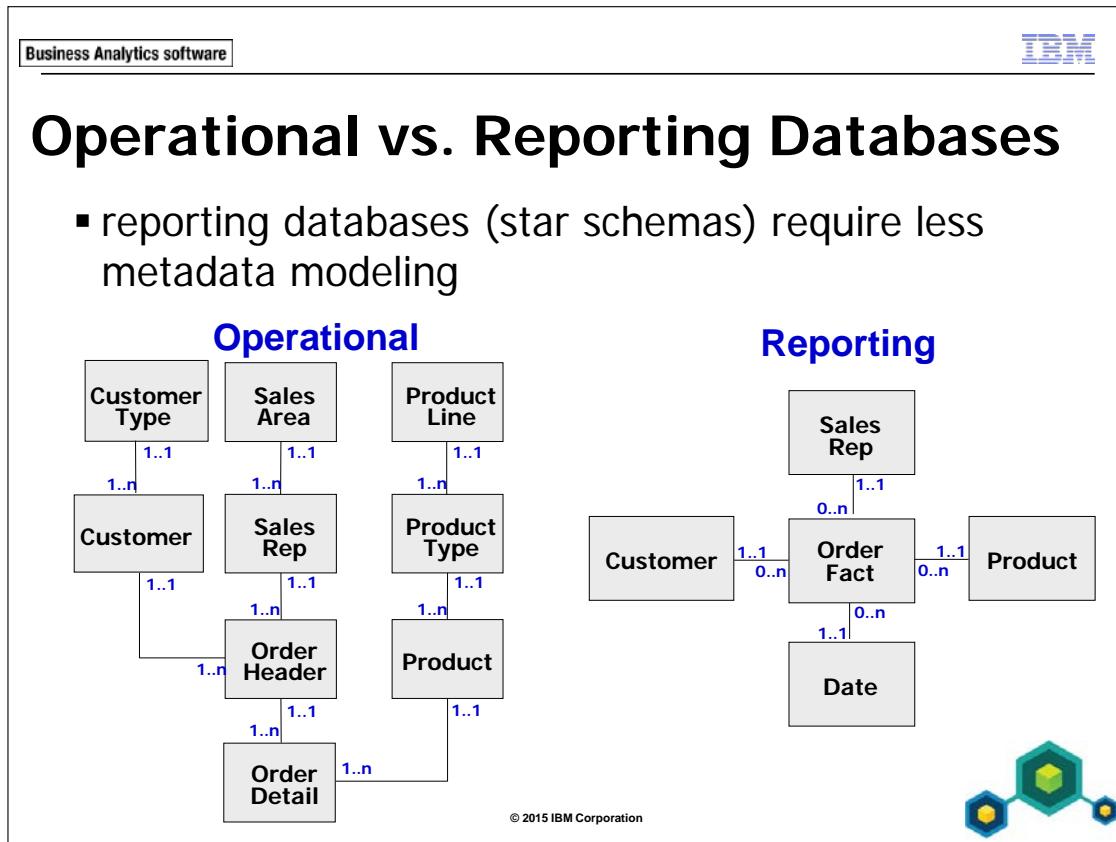
© 2015 IBM Corporation



When you modify a data source query subject, it may be considered a "complex" query subject rather than a simple query subject that is based on a simple, all-inclusive select statement. In this case, Cognos may need to make extra calls to the database for metadata at run time to fulfill the query request. This issue and resolutions to it are addressed in the Optimize and Tune Framework Manager Models module.

When creating model query subjects with relationships attached, the generated SQL will honor any encapsulated join syntax for that query subject. All other underlying relationships that are not encapsulated, are no longer accessible. Modelers should be aware of this and understand that this is expected behavior to give them a technique in which they can control query paths and obtain consistent and predictable results.

Modelers should always review the generated SQL of any query subject to be used in the final business view, especially, if they have modified them. This is so they can determine that Cognos is generating the SQL they expected to achieve the desired results.



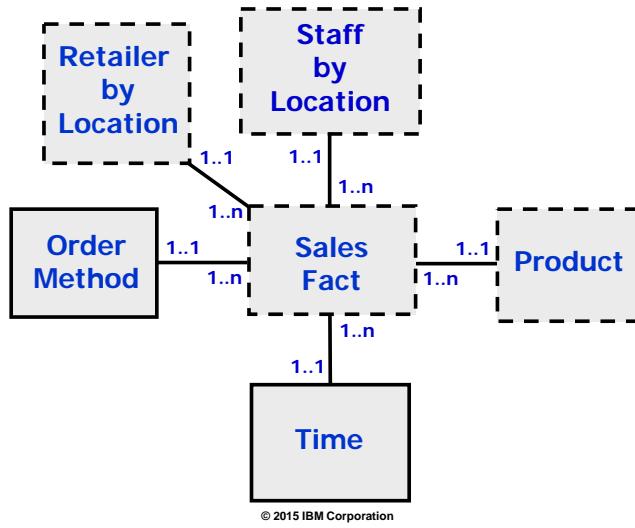
Operational databases typically require more of a metadata modeling effort in Framework Manager since they are confusing for users to understand and can present several reporting traps.

This course focuses on modeling an operational source because it illustrates more modeling scenarios that may apply to any database, both operational and reporting.

Star schema databases will likely require much less modeling since they are already designed for reporting. However, not all reporting database designs are immune to design issues and may require some of the very same techniques found in this course.

Modeling Virtual Star Schemas

- Model operational metadata as a virtual star schema to control SQL generation and provide predictable results.



In the slide example, query subjects with dashed borders represent model query subjects that have been used to create views of the data at run time. These views organize and/or control the SQL generation, which helps to provide an easy to use model and predictable results.

When you model an operational or reporting data source as a star schema, you are not changing the underlying, data structure. You are creating a virtual star schema that allows IBM Cognos to generate the correct SQL for predictable results. Another benefit of virtual star schema models is that they are more flexible, allowing authors to answer a wider range of queries.

Why Model as a Star Schema?

- lets you create star schema groupings (packages)
- simpler to understand because there are fewer query subjects
- adaptable and extendable: you can easily add and reuse facts and dimensions
- conformed dimensions prevent data silos: facts are related to one another through dimensions

© 2015 IBM Corporation



The advantage of modeling dimensionally is that the end result is a business view that is organized by subject area. By presenting each subject area in a namespace with the relevant objects, it is easier for a business user to select the appropriate dimensions to go with a particular fact. Also, adding more facts and relevant dimensions can extend the model without affecting the existing metadata presentation. This allows you to easily add another subject area to the model.

Note that some people, with a relational database design background, challenge the idea of modeling operational data as a star schema because it generates lots of SQL at the reporting end. This is true, but it is an accepted trade-off to present authors with a structure that ensures that they retrieve the correct data. If you want simpler SQL in the reports without the risk of queries retrieving incorrect data (a risk encountered when you report directly on operational data structures), your only other solution is to physically store the data in a star schema.

Benefits of Using Model Query Subjects

- Use model query subjects to:
 - prevent ambiguous joins
 - specify joins between multiple query subjects ("as view" behavior)
 - simplify the model for presentation
 - override relationship settings for underlying query subjects
 - resolve recursive relationships

© 2015 IBM Corporation



Model query subjects simply take a copy of the characteristics of the underlying objects on which they are based when they are created. They are independent of the underlying objects and can have their default behaviors changed without affecting the underlying objects. Changes to underlying objects will also not be reflected in the model query subjects. The query engine interprets the model query subject to be a fact or dimension, based on the cardinalities you attach to it.

You can use model query subjects to:

- Control query paths by using model query subjects as aliases to prevent ambiguous join paths, when multiple potential paths exist.
- Group query items from multiple query subjects to create a simplified Business view. For example, you can combine product line, product type and product info into a simplified and more intuitive product dimension.
- Resolve recursive relationships by creating an alias of a query subject and then relating it back to itself.

Note: After implementing a relationship, you cannot also rely on underlying relationships to other query subjects. At that point you must implement all relationships required for queries with that model query subject.

Business Analytics software

IBM

Using Model Query Subjects to Override Relationships

- you can use model query subjects to override relationship settings

Model

Query Subjects

```

graph LR
    Employee[Employee] -- "1..1" -->|dashed red circle| SalesTargetFact[Sales Target Fact]
    
```

Data Source

Query Subjects

```

graph LR
    EMPLOYEE[EMPLOYEE] -- "1..1" -->|red line| SALES_TARGET_FACT[SALES_TARGET_FACT]
    
```

© 2015 IBM Corporation

Using model query subjects, you can override underlying relationships to meet alternate reporting requirements. For example, in some reporting instances, authors would like to report only on employees who have sales targets. The data source query subjects' relationship meets this requirement because they have an inner join. But for other authors who would like to report on all employees regardless of whether they have sales targets or not, they can use the model query subjects to accomplish this since the cardinality has been changes to optional on the Sales Target Fact side.

Again, once you begin attaching relationships to model query subjects, you will need to create all required relationships for the query subject to meet your reporting needs. For example, if Sales Target Fact in the slide example will be queried with Time, then a relationship to Time would need to be created. The IBM Cognos query engine would not go back down to the data source query subject level to try and use the original relationship between SALES_TARGET_FACT and TIME_DIMENSION.

You can also use shortcut to override underlying relationships.

Demo 1: Use Model Query Subjects to Resolve Ambiguous Query Paths

Purpose:

You want to review the model to find any ambiguous joins, and use model query subjects to resolve any issues that you identify.

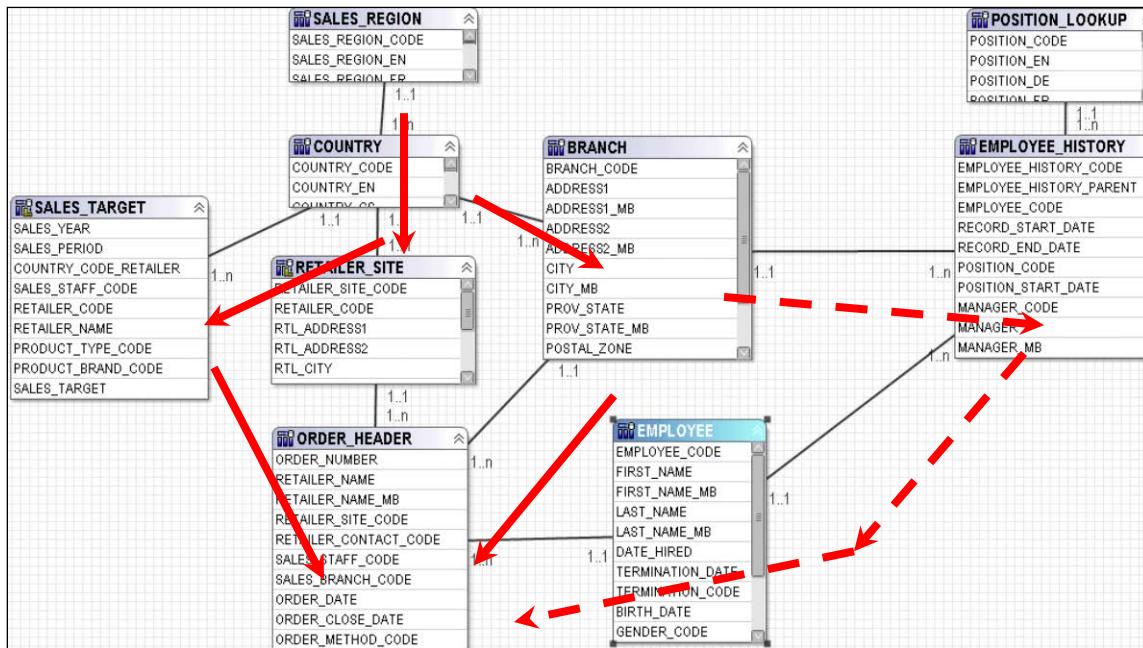
Component: Framework Manager

Project: GO Operational

Task 1. Review the ambiguous join reporting trap.

1. In **Framework Manager**, open the **GO Operational** model located at **C:\Edcognos\B5A52\CBIFM-Start Files\Module 7\GO Operational**. If prompted, log in as User ID **admin**, and Password **Education1**.
2. In the **Project Viewer** pane, in the **gosales** namespace, right-click **COUNTRY**, and then click **Launch Context Explorer**.
3. If **Context Explorer** only displays the **COUNTRY** table, click **Show Related Objects** .
4. Click **BRANCH**, and then click **Show Related Objects**.
5. Click **EMPLOYEE_HISTORY**, and then click **Show Related Objects**.

Arrange the diagram similar to below, to better see the hierarchy structure:



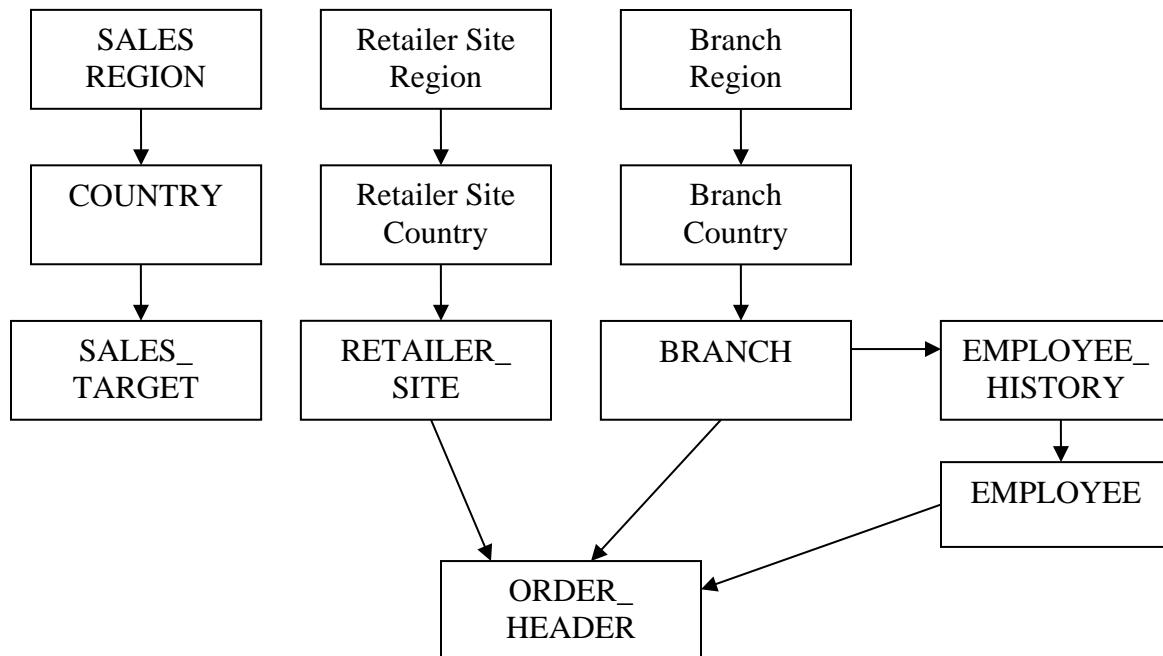
This material is meant for IBM Academic Initiative use only. NOT FOR RESALE

© 2003, 2015, IBM Corporation

This guide contains proprietary information which is protected by copyright. No part of this document may be photocopied, reproduced, or translated into another language without a legal license agreement from IBM Corporation.

SALES_REGION and COUNTRY have two separate paths that lead to ORDER_HEADER (three if you include the route through EMPLOYEE_HISTORY and EMPLOYEE). To resolve ambiguity, create three aliases of SALES_REGION and COUNTRY, and link each to one of the paths to ORDER_HEADER and ORDER_DETAIL. Keep the original SALES_REGION and COUNTRY to link to SALES_TARGET.

Note: You will achieve the following design at the end of this demo:



6. Close the **Context Explorer**.

Task 2. Create Region aliases as model query subjects.

1. In the **Project Viewer** pane, right-click **gosales**, point to **Create**, and then click **Query Subject**.
2. Name the query subject **Retailer Site Region (alias)**, and then with **Model** selected, click **OK**.
3. Expand **Foundation Objects View > gosales > SALES_REGION**.

4. Add the following query items by double-clicking them.

SALES_REGION_CODE

SALES_REGION_EN

This alias could also have been created by expanding SALES_REGION in the Project Viewer, selecting the two query items, right-clicking one of them, and clicking Merge in New Query Subject. Both methods have the same results.

You will include only the columns your report authors will require. Later in the course, you will replace SALES_REGION_EN with a calculation that dynamically picks up the appropriate language column based on the language setting of the user's computer.

You should rename the items to be more user-friendly before you make copies.

5. Click **OK**, and then in the **Project Viewer** pane, rename the query items as follows:

- **Retailer Site Region Code**
- **Retailer Site Region**

6. Right-click **Retailer Site Region (alias)**, and select **Edit > Copy**.

7. Right-click the **gosales** namespace, and select **Edit > Paste**.

8. Rename the copy, and its two query items, as follows:

- **Branch Region (alias)**
 - **Branch Region Code**
 - **Branch Region**

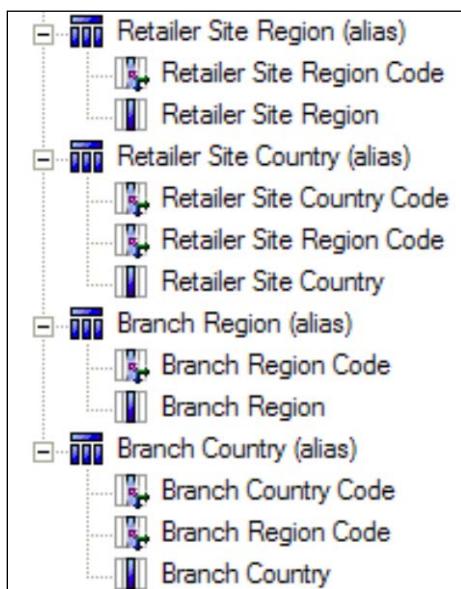
Task 3. Create Country aliases as model query subjects.

1. Repeat Task 2 to create two aliases of **COUNTRY** as follows:

- **Retailer Site Country (alias)**
 - **Retailer Site Country Code** (from COUNTRY.COUNTRY_CODE)
 - **Retailer Site Region Code** (from SALES_REGION.SALES_REGION_CODE)
 - **Retailer Site Country** (from COUNTRY.COUNTRY_EN)
- **Branch Country (alias)**
 - **Branch Country Code**
 - **Branch Region Code**
 - **Branch Country**

2. Drag **Retailer Site Country (alias)** below **Retailer Site Region (alias)**.

The results appear as follows:



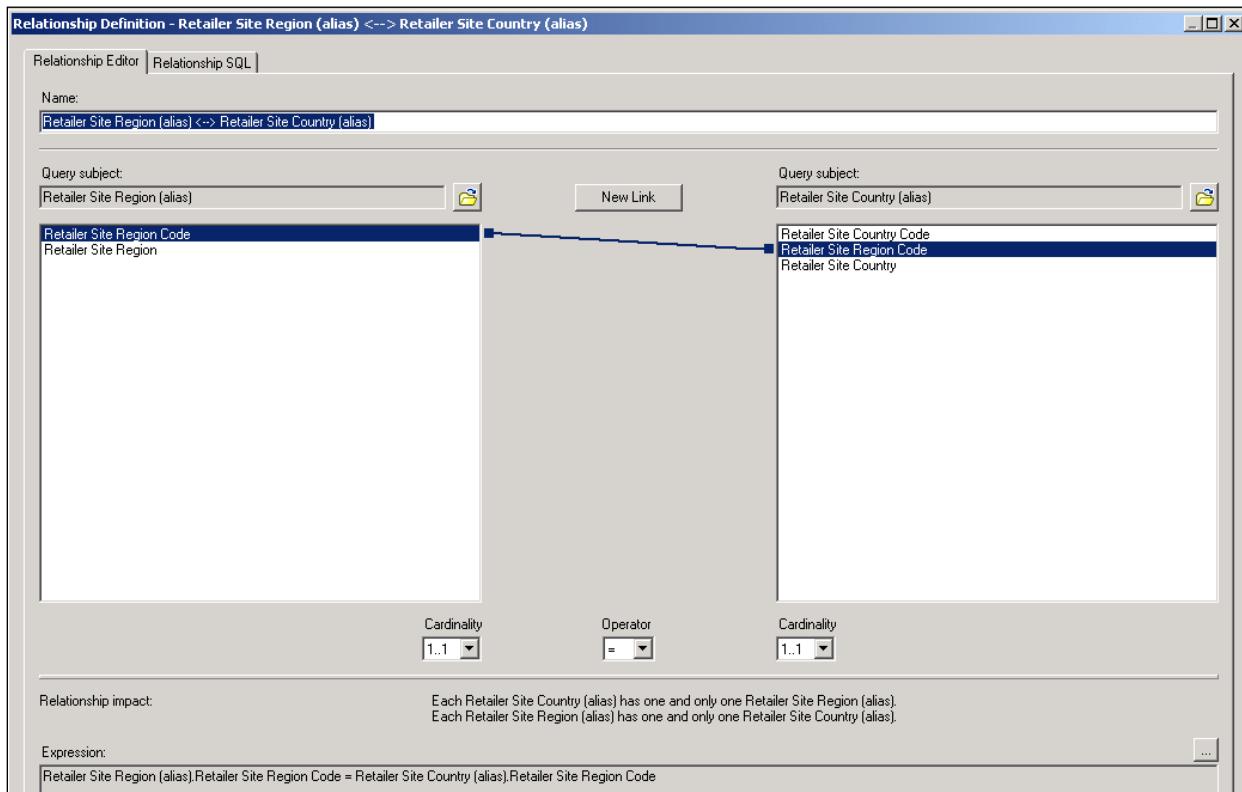
You now have aliases for two of the three relationships between Region/Country and ORDER_HEADER/ORDER_DETAILS. You will create the third one (for employees) in a workshop.

Note: The original SALES_REGION and COUNTRY query subjects will be used for relationships to SALES_TARGET later in the course. You will not rename them now because they are original data source query subjects, not model query subjects.

Task 4. Create relationships.

1. In the **Project Viewer** pane, select the following items:
 - **Retailer Site Region (alias) > Retailer Site Region Code**
 - **Retailer Site Country (alias) > Retailer Site Region Code**
2. Right-click one of the selected items, point to **Create** and then click **Relationship**.

The results appear as follows:



You now have a relationship between the two query subjects on a common key with the correct cardinality. Again, using this method, the first object selected will be on the 1..1 side of the relationship. You can modify the cardinality after the fact if it does not meet your needs.

3. Under **Retailer Site Country (alias)**, set the **Cardinality** to **1..n**.
4. Click **OK**.
5. Click **No**, when prompted about existing underlying relationships.

6. Repeat the above steps to create a relationship from **Branch Region (alias)** (1..1) to **Branch Country (alias)** (1..n) on **Branch Region Code**.

As shown in the diagram in Task 1, you also want to create relationships between the COUNTRY aliases and the query subjects that lead them to ORDER_HEADER, RETAILER_SITE and BRANCH respectively.

7. Repeat steps **1** to **5** to create the following relationships:

- **Retailer Site Country (alias)** (Retailer Site Country Code, 1..1) to **RETAILER_SITE** (RTL_COUNTRY_CODE, 1..n)
- **Branch Country (alias)** (Branch Country Code, 1..1) to **BRANCH** (COUNTRY_CODE, 1..n)

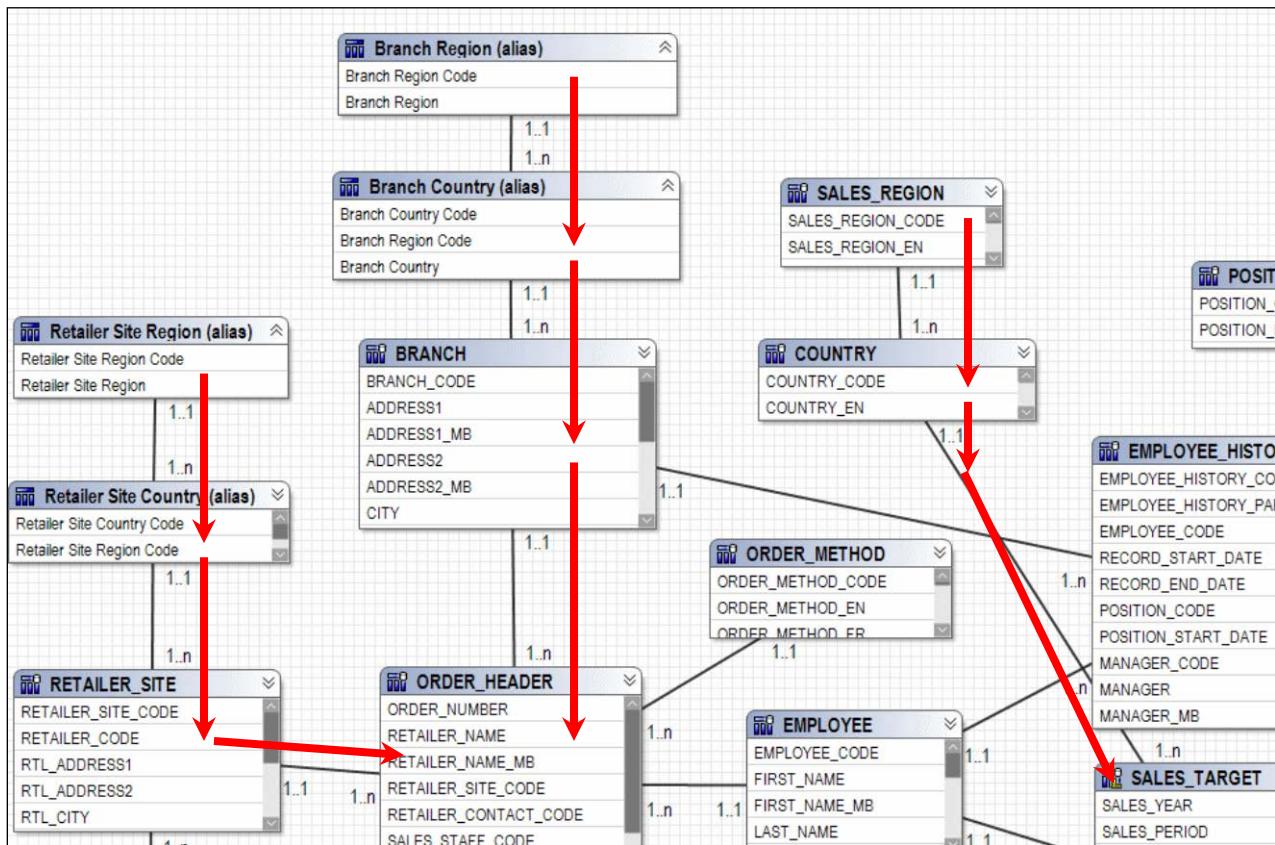
Task 5. Remove old relationships.

1. Right-click **COUNTRY**, click **Launch Context Explorer**, and then click **Show Related Objects**.

Next, you will remove relationships that are no longer required. In specific, COUNTRY should no longer link to RETAILER_SITE or BRANCH, now that Retailer Site Country (alias) links to RETAILER_SITE and Branch Country (alias) links to BRANCH.

2. Delete the relationship between the following items:
 - **COUNTRY** and **RETAILER_SITE**
 - **COUNTRY** and **BRANCH**
3. Close **Context Explorer**.
4. In the center pane, click **Diagram**.

5. From the **Diagram** menu, click **Diagram Settings**, under **Level of Details**, select **Query Items**, and then click **OK**.
6. In the **Diagram**, arrange the objects as shown below:

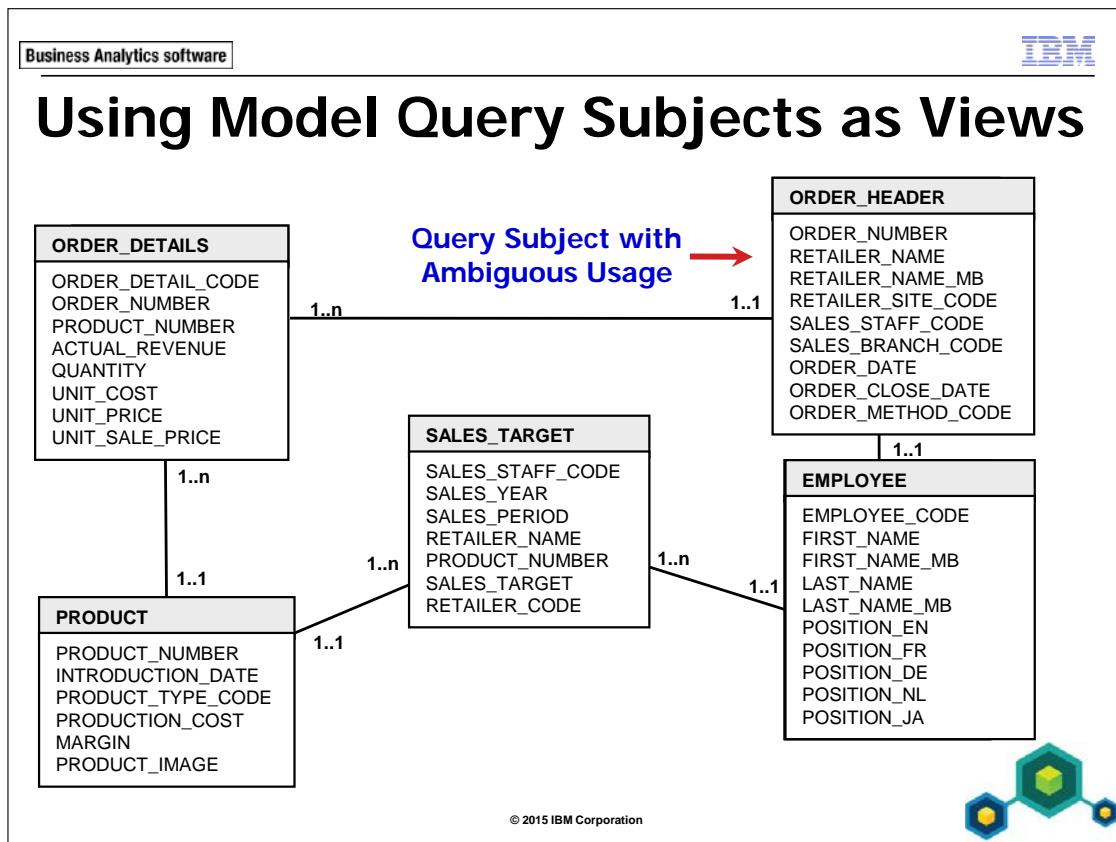


SALES_REGION and COUNTRY no longer have ambiguous query paths. There is now a clear path for retailer queries, branch queries and sales target queries.

7. Save the project and leave Framework Manager open for the next demo.

Results:

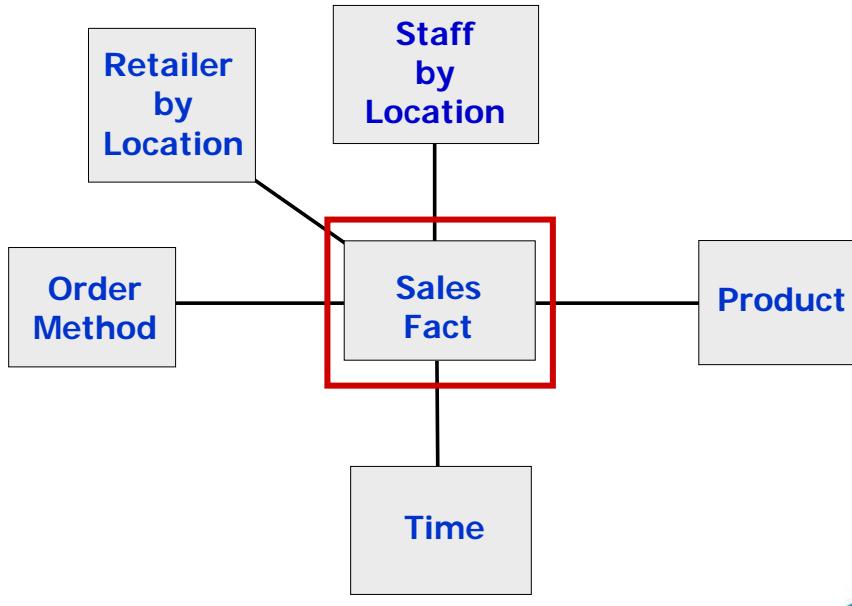
You have taken the first steps to resolving a situation where the proper relationship between SALES_REGION/COUNTRY and ORDER_HEADER/ORDER_DETAILS could be ambiguous. You have created multiple copies of the ambiguous query subjects and created the required relationships among them.



Because of the mix of cardinalities, the query engine could identify **ORDER_HEADER** as either a fact or a dimension, depending on the query in which it is used. To remove this ambiguity, you can merge **ORDER_HEADER** and **ORDER_DETAILS** into a model query subject, keeping only required query items.

By merging **ORDER_HEADER** and **ORDER_DETAILS** and maintaining existing relationships, you create "As View" behavior in which the underlying join between them is always honored. In this way, these two query subjects will always be treated as one, and in this case, always as a fact.

Requirements Review: Sales Fact



© 2015 IBM Corporation



In the next demo, you will merge ORDER_HEADER and ORDER_DETAILS into a model query subject in order to meet your Sales Fact requirement.

Demo 2: Create a Sales Fact Model Query Subject

Purpose:

To further simplify the model, you will create a **Sales Fact** model query subject that merges **ORDER_HEADER** and **ORDER_DETAILS** into one model query subject. This ensures that the query path through **ORDER_HEADER** is maintained when reporting on sales facts, and that the query engine correctly interprets the usage.

Component: **Framework Manager**

Project: **GO Operational**

Task 1. Create a Sales Fact model query subject.

1. In the **Project Viewer** pane, select **ORDER_HEADER** and **ORDER_DETAILS**, right-click one of the selected items, and then click **Merge in New Query Subject**.

You are asked if you would like to recreate existing relationships.

2. Click **Yes**.

You want all the relationships recreated because this is a fact query subject that connects to many dimensional query subjects. You did not want to recreate relationships when creating aliases earlier, because you were specifically creating new relationships to avoid ambiguous query paths.

By merging these two objects and maintaining relationships, the query subject will behave as a view and always generate the underlying join between the two query subjects.

3. Rename the new model query subject to **Sales Fact**.
4. Double-click **Sales Fact** to open its **Query Subject Definition** dialog box.

5. Click, and then click **Delete**, for each of the following query items:

RETAILER_NAME
RETAILER_NAME_MB
ORDER_NUMBER1

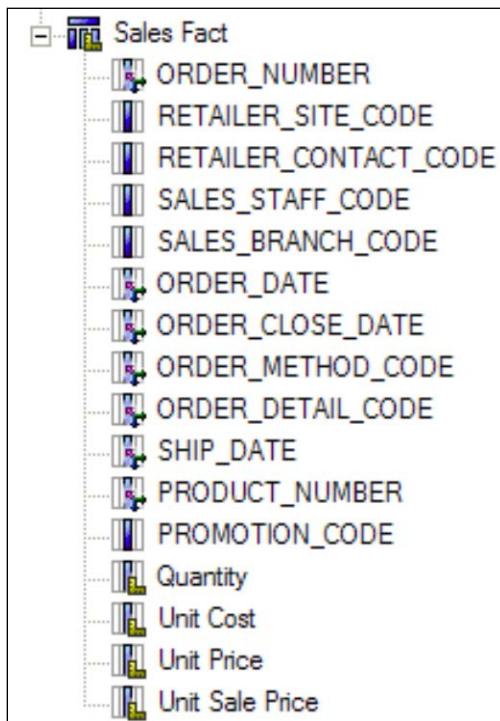
By deleting the retailer names, you are preventing authors from accessing the incorrect version of the retailer name and inadvertently taking descriptive information from a fact query subject.

ORDER_NUMBER1 is a redundant query item merged in from **ORDER_DETAILS**. You will use **ORDER_NUMBER** from **ORDER_HEADER**.

6. Click **OK**, and then rename:

- **QUANTITY** to **Quantity**
- **UNIT_COST** to **Unit Cost**
- **UNIT_PRICE** to **Unit Price**
- **UNIT_SALE_PRICE** to **Unit Sale Price**

The results appear as follows:



These are the query items that will eventually be used in the Presentation View which your authors will see.

Task 2. Test "As View" behavior.

1. Test the **Quantity** query item, and then view the **Query Information** tab.
The results appear as follows:

```
Cognos SQL
select
    Sales_Fact.Quantity as Quantity
from
    (select
        ORDER_DETAILS.QUANTITY as Quantity
    from
        GOSALES..GOSALES.ORDER_HEADER ORDER_HEADER,
        GOSALES..GOSALES.ORDER DETAILS ORDER DETAILS
    where
        (ORDER HEADER.ORDER NUMBER = ORDER DETAILS.ORDER NUMBER)
) Sales_Fact
```

Although you selected an item only associated with ORDER_DETAILS, a join with ORDER_HEADER is honored as seen in the where clause. The two underlying query subjects are now treated as one.

2. Click **Close**.

You will now test the behavior of a merged query with no relationships attached.

3. In the **Project Viewer** pane, select **ORDER_HEADER** and **ORDER_DETAILS**, right-click one of the selected items, and then click **Merge in New Query Subject**.

You are asked if you would like to recreate existing relationships.

4. Click **No**.

5. Expand **ORDER_HEADER_ORDER_DETAILS**, test **QUANTITY**, and then click the **Query Information** tab.

The results appear as follows:

```
Cognos SQL
select
    ORDER_DETAILS.QUANTITY as QUANTITY
from
    GOSALES..GOSALES.ORDER_DETAILS ORDER_DETAILS
```

There is no join with ORDER_HEADER. This new query subject is simply acting as a container for the two underlying query subjects. Since you only queried an item from ORDER_DETAILS, the SQL is minimized and only requests information from that one table in the database.

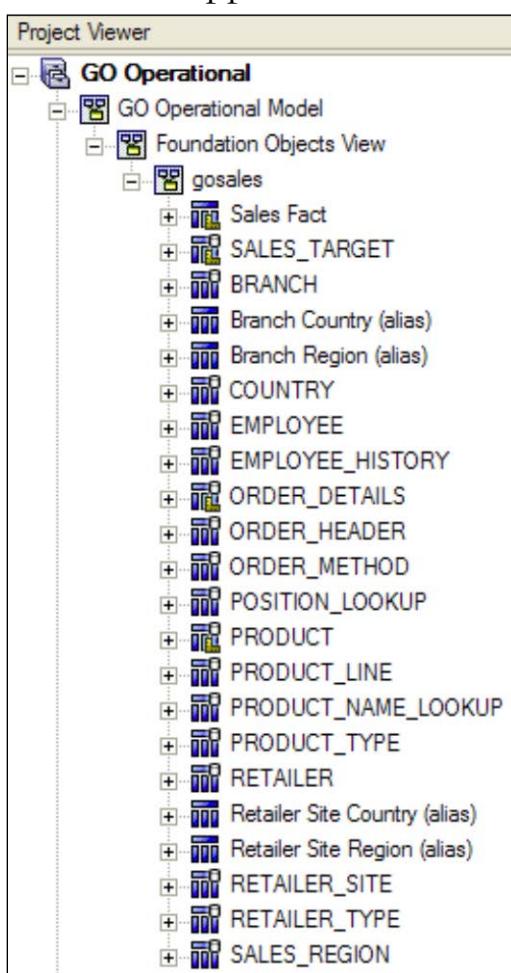
6. Click **Close**, and then click **Undo** to remove the new query subject.

Task 3. Organize gosales namespace and reduce model clutter.

As a convention, you will sort your project objects alphabetically, but place all facts before all dimensions.

1. In the **Project Viewer**, right-click the **gosales** namespace, and then click **Reorder**.
2. Ensure **Ascending** is selected, and then click **OK**.
3. Drag **Sales Fact** below the **gosales** namespace, and then drag **SALES_TARGET** below **Sales Fact**.

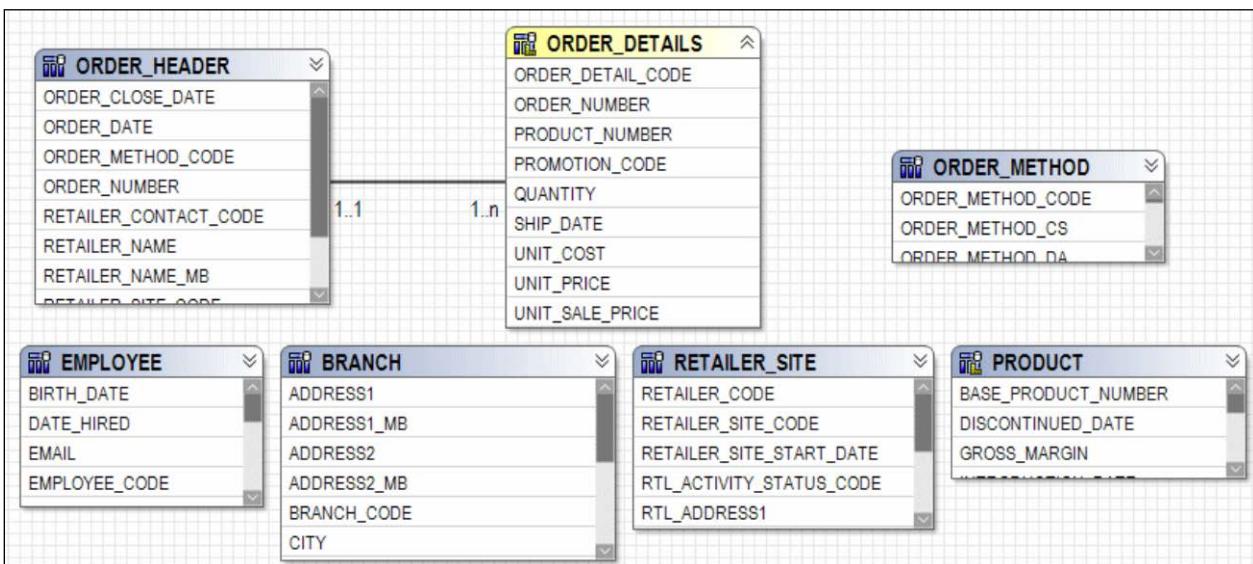
The results appear as follows:



Original data source query subjects that have been replaced by model query subjects (in order to prevent reporting traps) will be placed in a separate folder and non-required relationships will be deleted in order to reduce model clutter. You will do this now for ORDER_HEADER and ORDER_DETAILS.

4. Right-click the **gosales** namespace, point to **Create**, click **Folder**, name it **Original Sales Objects**, and then click **Next**.
5. Expand **Foundation Objects View > gosales**, and select the check boxes for **ORDER_DETAILS** and **ORDER_HEADER**.
They will change to green checks.
6. Click **Finish**.
7. In the **Project Viewer**, expand **Original Sales Objects**.
8. Ctrl+click to select **ORDER_HEADER** and **ORDER_DETAILS**, right-click one of the selected items, and then click **Launch Context Explorer**.
9. Click **Show Related Objects**, and then delete all relationships except the one between **ORDER_HEADER** and **ORDER_DETAILS**.

The results appear similar to the following:



10. Close the **Context Explorer**.
11. Save your project.

Task 4. Test your new virtual star schema query subjects (optional).

In the Ambiguous Relationships demo in the Identifying Reporting Issues module, there were multiple paths between COUNTRY_EN and QUANTITY, so there was potential for unexpected results. Now authors must explicitly take the country name from either Retailer Country (down one path) or Branch Country (down another path), depending on whether they want the countries of the clients (retailers) or of the sales offices (branches). The results are always expected and predictable.

- Select and **Test** the following query items with **Auto Sum** enabled:

Query Subject	Query Item
Retailer Site Country (alias)	Retailer Site Country Code
Sales Fact	Quantity

The goal is to report on the number of units sold to each retailer's country.

- Click the **Query Information** tab.

The results appear as follows:

```
Cognos SQL
select
    Retailer_Site_Country_alias_.Retailer_Site_Country_Code as Retailer_Site_Country_Code,
    XSUM(Sales_Fact.Quantity for Retailer_Site_Country_alias_.Retailer_Site_Country_Code ) as Qua
from
    (select
        COUNTRY.COUNTRY_CODE as Retailer_Site_Country_Code
        from
            GOSALES..GOSALES.COUNTRY COUNTRY
    ) Retailer_Site_Country_alias_,
    (select
        ORDER_HEADER.RETAILER_SITE_CODE as RETAILER_SITE_CODE,
        ORDER_DETAILS.QUANTITY as Quantity
        from
            GOSALES..GOSALES.ORDER_HEADER ORDER_HEADER,
            GOSALES..GOSALES.ORDER_DETAILS ORDER_DETAILS
        where
            (ORDER_HEADER.ORDER_NUMBER = ORDER_DETAILS.ORDER_NUMBER)
    ) Sales_Fact,
    GOSALES..GOSALESRT.RETAILER SITE RETAILER SITE
where
    (Retailer_Site_Country_alias_.Retailer_Site_Country_Code = RETAILER_SITE.RTL_COUNTRY_CODE) and
    (RETAILER SITE.RETAILER SITE CODE = Sales_Fact.RETAILER SITE CODE)
group by
    Retailer_Site_Country_alias_.Retailer_Site_Country_Code
```

Notice the final where clause. The query takes the path you expected, Retailer Country (alias) to RETAILER_SITE to Sales Fact. This query works because the model has no alternative query paths. You have removed any ambiguity. The same concept would apply if you tested Branch Country (alias) and Sales Fact.

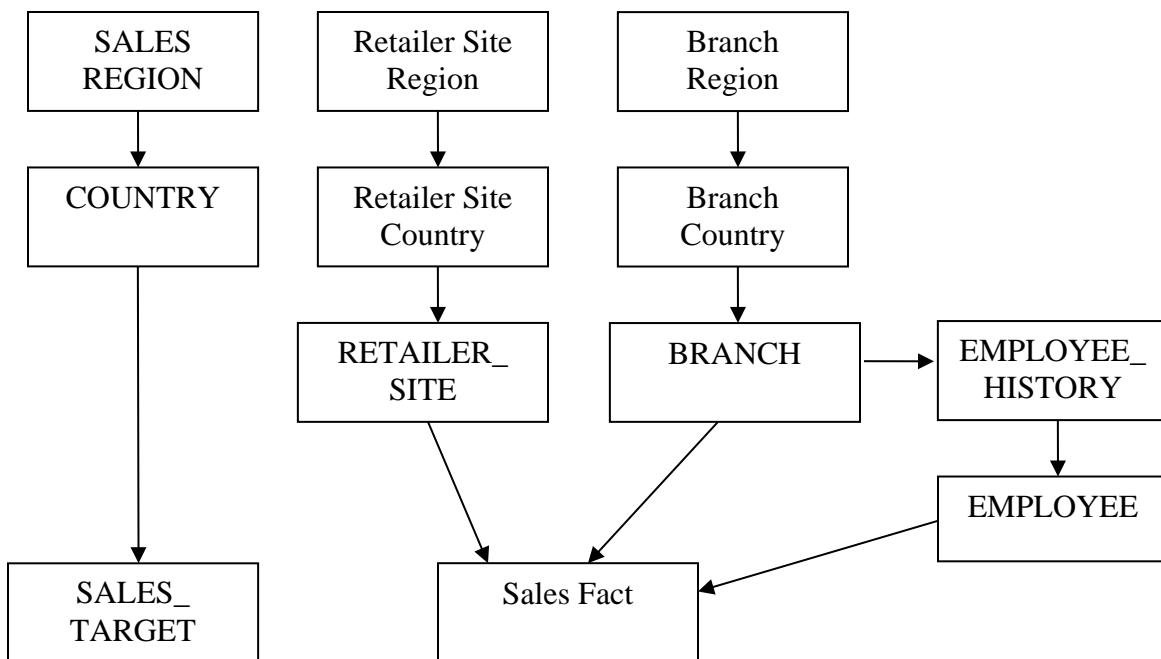
3. Close the test results window, save the project, and leave Framework Manager open for the Workshop.

Results:

To remove ambiguity that may occur with the ORDER_HEADER data source query subject, you merged ORDER_HEADER with ORDER_DETAILS. You then organized your model and reduced clutter and tested your new alias query subjects with the new Sales Fact query subject to ensure ambiguity has been removed.

Workshop 1: Create Staff-Related Model Query Subjects

In the last two demos, you wanted report authors to avoid accidentally choosing the wrong path between SALES_REGION/COUNTRY and ORDER_HEADER/ORDER_DETAILS, so you created two aliases of SALES_REGION and COUNTRY, one for retailer sites and one for branches. You also, merged ORDER_HEADER and ORDER_DETAILS into a Sales Fact model query subject.



However, BRANCH links to Sales Fact in two paths; directly, as well as through EMPLOYEE_HISTORY and EMPLOYEE.

Workshop 1: Tasks and Results

Task 1. Create Staff Region (alias) and Staff Country (alias).

- Create a third alias for **SALES_REGION** and **COUNTRY** as follows:
 - **Staff Region (alias)**
 - **Staff Region Code** (from SALES_REGION_CODE)
 - **Staff Region** (from SALES_REGION_EN)
 - **Staff Country (alias)**
 - **Staff Country Code** (from COUNTRY_CODE)
 - **Staff Region Code** (from SALES_REGION_CODE)
 - **Staff Country** (from COUNTRY_EN)

Tip: to quickly create the alias tables, copy and paste Retailer Site Region (alias), and Retailer Site Country into the gosales namespace, and then rename the objects.

Task 2. Create relationship.

- Create a relationship between **Staff Region (alias)** (**Staff Region Code**, 1..1) and **Staff Country (alias)** (**Staff Region Code**, 1..n).

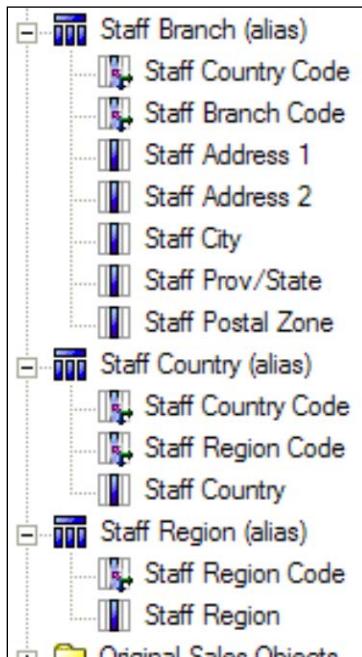
Note: Do not replicate underlying relationships.

You also need a relationship from Staff Country (alias) to BRANCH, but Branch Country (alias) already has a join to BRANCH. To avoid ambiguity, you first need to create an alias for BRANCH, named Staff Branch (alias) to indicate its purpose.

Task 3. Create Staff Branch (alias) model query subject.

- Create a new model query subject called **Staff Branch (alias)**, with the following query items (taken from **BRANCH** and renamed):
 - Staff Country Code
 - Staff Branch Code
 - Staff Address1
 - Staff Address2
 - Staff City
 - Staff Prov/State
 - Staff Postal Zone
- Move **Staff Branch (alias)** above **Staff Region (alias)** in the **Project Viewer**.

Your Foundation Objects View should include the new objects shown below:

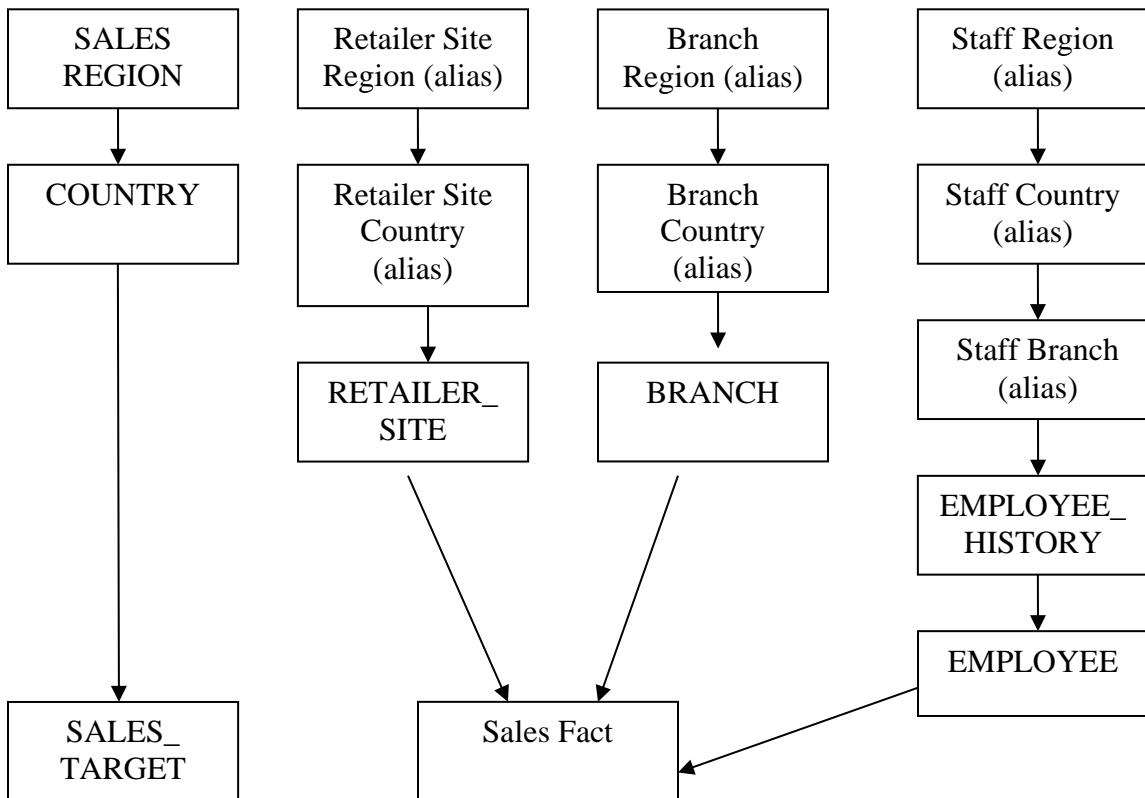


Task 4. Create branch relationships.

- Create a relationship between **Staff Country (alias)** (based on **Staff Country Code, 1..1**) and **Staff Branch (alias)** (based on **Staff Country Code, 1..n**), and do not replicate underlying relationships.
- Create a relationship between **Staff Branch (alias)** (based on **Staff Branch Code, 1..1**) and **EMPLOYEE_HISTORY** (based on **BRANCH_CODE, 1..n**), and do not replicate underlying relationships.

Task 5. Delete an unwanted relationship and organize the project.

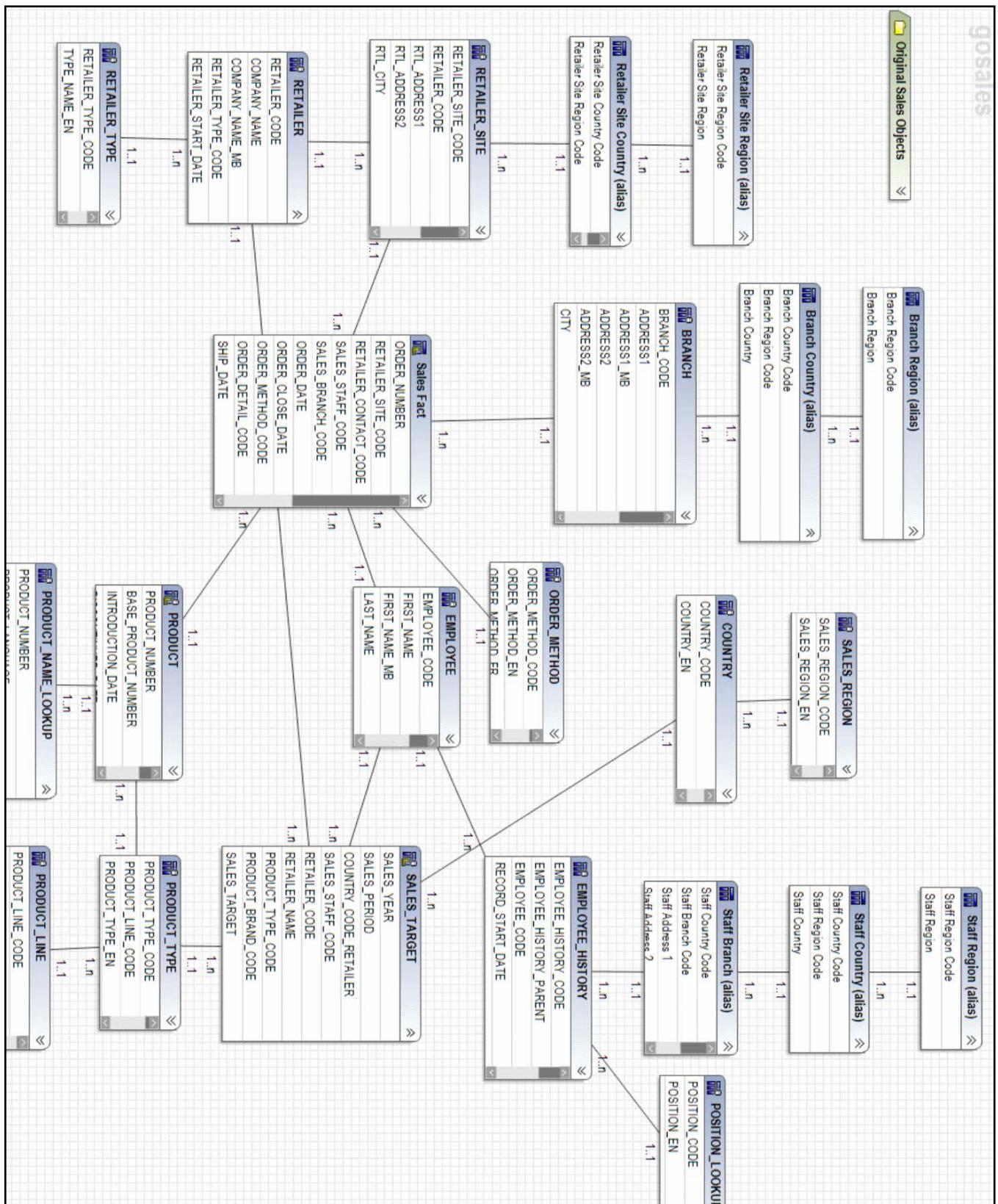
- Delete the relationship between the original **BRANCH** and **EMPLOYEE_HISTORY**.
- In the **Project Viewer**, sort the new objects alphabetically.
- In the **Diagram**, arrange the diagram to show the new Staff query path from Staff Region through to **EMPLOYEE**:



- Save the project, and leave Framework Manager open for the next workshop.

At the end of the workshop, the results appear as follows:

Your Diagram displays the new staff query path.



This material is meant for IBM Academic Initiative use only. NOT FOR RESALE

© 2003, 2015, IBM Corporation

© 2003, 2013, IBM Corporation.
This guide contains proprietary information which is protected by copyright. No part of this document may be photocopied, reproduced, or translated into another language without a legal license agreement from IBM Corporation.

Workshop 2: Merge Query Subjects to Control Usage

Based on the modeling in the freehand exercise, you found that both **PRODUCT_TYPE** and **RETAILER** could be interpreted as either a fact or a dimension.

To remove this ambiguity, you will merge them with related query subjects.

- Merge **PRODUCT_TYPE** with **PRODUCT** (recreating relationships), rename the new query subject **Product Type & Product**, and remove any redundant query items.

Move original query subjects into a new folder called **Original Product Objects** and delete all relationships to them except the one between each other.

- Merge **RETAILER** with **RETAILER_SITE** (recreating relationships), rename the new query subject to **Retailer & Retailer Site**, and remove any redundant query items.

Move original query subjects into a new folder called **Original Retailer Objects** and delete all relationships to them except the one between each other.

- Arrange the new model query subjects alphabetically.
- **Save** and the project.

For more information about where to work and the workshop results, refer to the Tasks and Results section that follows. If you need more information to complete a task, refer to earlier demos for detailed steps.

Workshop 2: Tasks and Results

Task 1. Merge product query subjects.

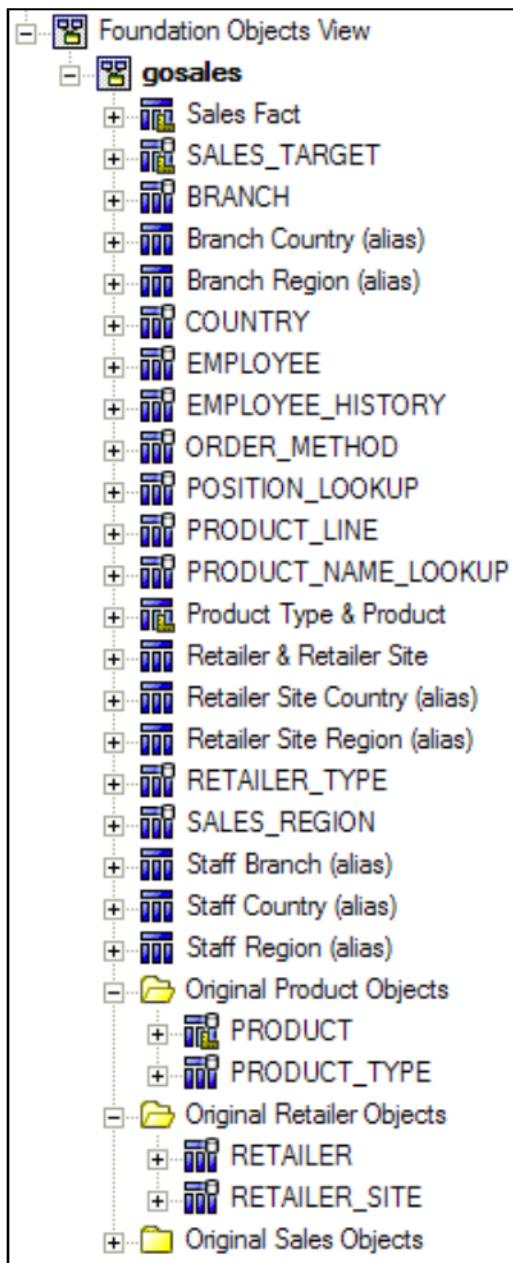
- In the **Project Viewer**:
 - merge **PRODUCT_TYPE** and **PRODUCT** into a new query subject, and click **Yes** to recreate relationships.
 - rename the merged query **Product Type & Product**
 - in the new query subject, delete **PRODUCT_TYPE_CODE1**
 - in **gosales**, create a new folder named called **Original Product Objects**, and then move the **PRODUCT_TYPE** and **PRODUCT** query subjects into that folder.
- Use **Context Explorer**, and **Show Related Objects**, to delete all relationships to **PRODUCT_TYPE** and **PRODUCT** except the relationship between **PRODUCT_TYPE** and **PRODUCT**.
- Manually arrange the new model query subject alphabetically.

Task 2. Merge retailer query subjects.

- In the **Project Viewer**:
 - merge **RETAILER** and **RETAILER_SITE** (recreating relationships), and rename it to **Retailer & Retailer Site**
 - delete **RETAILER_CODE1**
 - in **gosales**, create a new folder called **Original Retailer Objects**, and move **RETAILER** and **RETAILER_SITE** to the folder.
- In **Context Explorer**, delete all relationships to **RETAILER** and **RETAILER_SITE**, except the relationship between **RETAILER** and **RETAILER_SITE**.

- Manually arrange new model query subject alphabetically.

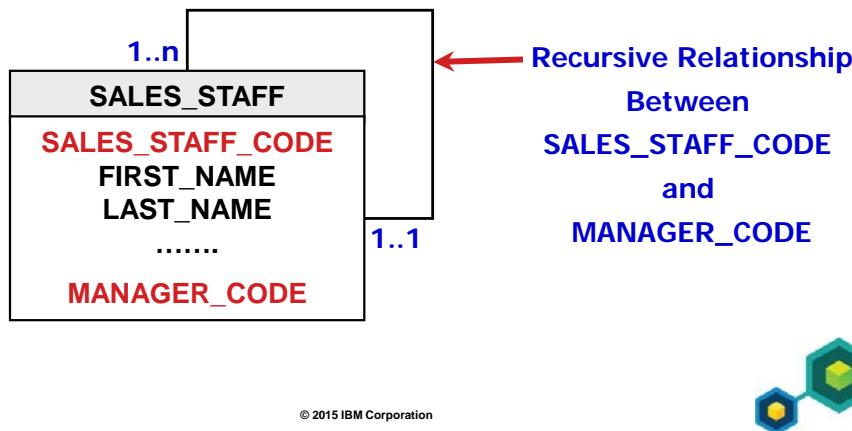
Your Foundation Objects View should include the new objects and folders shown below:



- **Save** the project and leave Framework Manager open for the next demo.

Identifying Recursive Relationships

- Framework Manager displays self-joins in the diagram, but does not execute them as queries.



You must specify the self-join relationship at the data source level for the recursive relationship to be displayed in Framework Manager.

While you can view the metadata that defines the relationship, you cannot edit a recursive relationship in Framework Manager.

Business Analytics software

IBM

Resolving Recursive Relationships

- Use model query subjects (or shortcuts) to create and modify recursive relationships.

The diagram shows two tables: 'Managers' and 'SALES_STAFF'. The 'Managers' table has columns 'SALES_STAFF_CODE', 'FIRST_NAME', and 'LAST_NAME'. The 'SALES_STAFF' table has columns 'SALES_STAFF_CODE', 'FIRST_NAME', 'LAST_NAME', and 'MANAGER_CODE'. A red arrow points from the 'SALES_STAFF_CODE' column in the 'Managers' table to the 'MANAGER_CODE' column in the 'SALES_STAFF' table. Relationship multiplicity is indicated as 1..1 for Managers to Sales_Staff and 1..n for Sales_Staff back to Managers. The text 'Recursive Relationship Can now be edited' is displayed below the tables.

© 2015 IBM Corporation

To modify a relationship that exists as a self-join in the data source, you can create a model query subject or shortcut and define a relationship between it and the original query subject. Using the two query subjects in the slide example, you can create a master-detail query based on the same table in the data source.

Using this solution, you can create a report that lists managers and all staff that report to them. In the following demo, the scenario will be slightly different and more complex than the one depicted in the slides.

Note: There is no actual recursive relationship on the database. This example shows a case where it could exist. Framework Manager does not recreate recursive relationships when metadata is imported, even if they actually exist in the database.

Demo 3: Resolve a Recursive Relationship

Purpose:

Currently the GO Operational project uses the MANAGER query item from the EMPLOYEE_HISTORY query subject to report on an employee's manager. Authors would like to show the managers' contact information and report on managers and their employees.

To accomplish this, you will create an alias of the EMPLOYEE data source query subject and relate it to the EMPLOYEE_HISTORY data source query subject on MANAGER_CODE, instead of EMPLOYEE code. This will allow you to use the EMPLOYEE table to retrieve manager names and their contact information.

Components: Framework Manager, IBM Cognos Workspace Advanced
 Project: GO Operational
 Package: GO Operational (query)

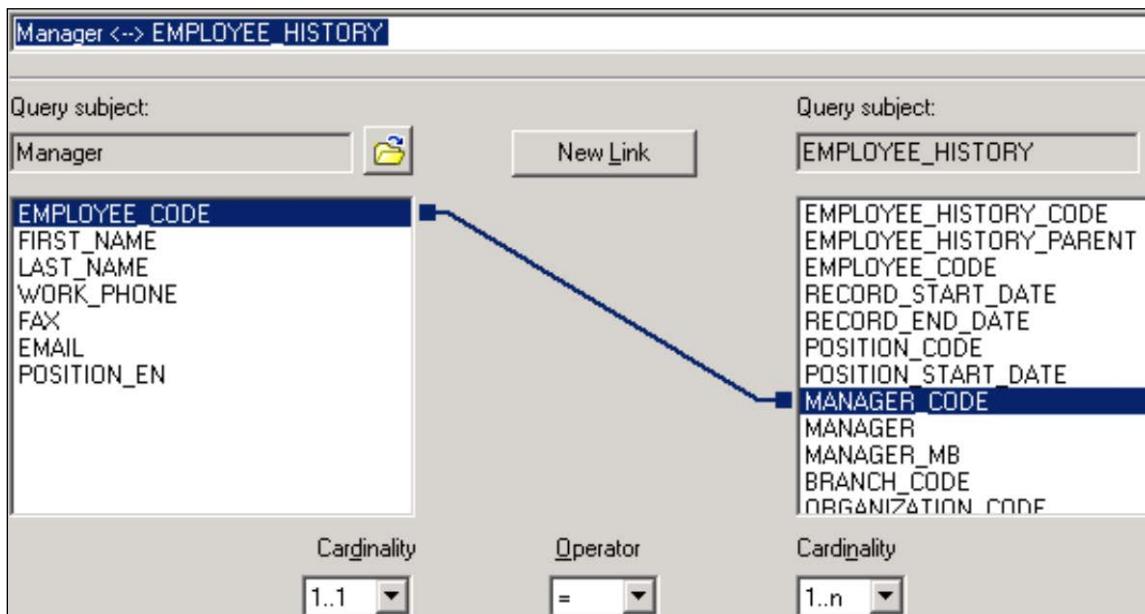
Task 1. Create an alias to resolve a recursive relationship in the data.

1. In **GO Operational Model > Foundation Objects View > gosales**, create a model query subject called **Manager**.
2. From the **Foundation Objects View**, add the following query items:

Query Subject	Query Item
EMPLOYEE	EMPLOYEE_CODE FIRST_NAME LAST_NAME WORK_PHONE EXTENSION FAX EMAIL
POSITION_LOOKUP	POSITION_EN

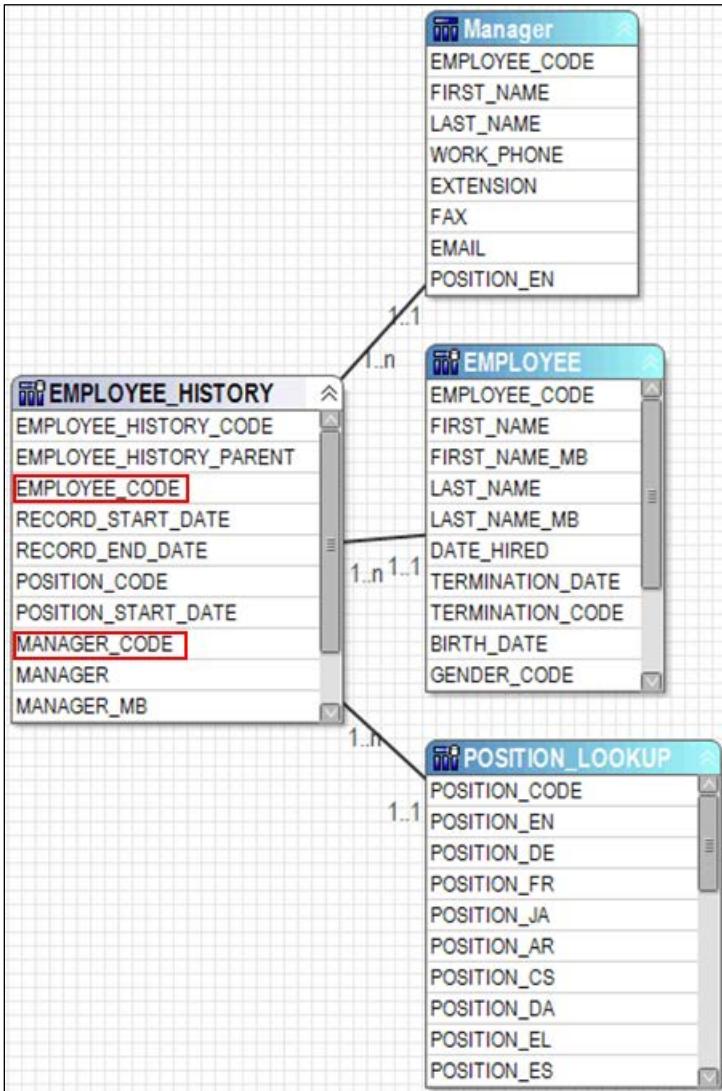
3. Click **OK**, and then create a relationship from **Manager (EMPLOYEE_CODE, 1..1)** to **EMPLOYEE_HISTORY (MANAGER_CODE, 1..n)**.

The results appear as follows:



4. Click **OK**, and then click **No** to replicating the existing relationships.
5. Move the **Manager** model query subject below **EMPLOYEE_HISTORY** to logically group the items together.
6. Right-click **EMPLOYEE_HISTORY**, and then click **Launch Context Explorer**.

7. Click the **Show Related Objects** button, click **Auto Layout**, if necessary, beside **Layout Style**, select **Standard**, and then set the distances to **25**.
The results appear as follows:

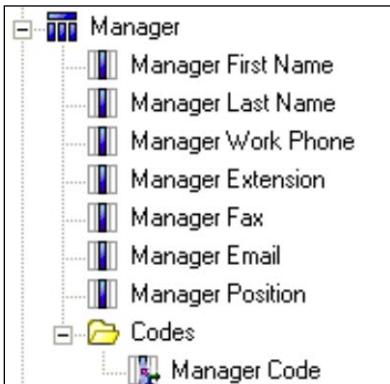


The recursive relationship is resolved, and you can use the Manager query subject to return data for managers and their staff. Because POSITION_EN is in the Manager model query subject (which now has a relationship defined) As View behavior will occur. All underlying relationships will be honored in any query that uses the Manager query subject.

8. Close the **Context Explorer**.

Task 2. Reorganize the query items and test the relationship.

1. In Manager, create a **Query Item Folder** named **Codes**, and move **EMPLOYEE_CODE** into it.
2. **Rename** and organize the query items as shown below:



3. In the **Project Viewer**, select and **Test**:
 - **EMPLOYEE.FIRST_NAME**
 - **EMPLOYEE.LAST_NAME**
 - **Manager.Manager First Name**
 - **Manager.Manager Last Name**

The results appear as follows:

Test results			
FIRST_NAME	LAST_NAME	Manager First Name	Manager Last Name
Jörg	Kunze	Denis	Pagé
Jeanne	Bertrand	Denis	Pagé
Gunter	Erler	Elsbeth	Wiesinger
Georg	Schmuker	Elsbeth	Wiesinger
Dieter	Vertemati	Elsbeth	Wiesinger
Dieter	Bakker	Elsbeth	Wiesinger
Lea	Schmidt	Elsbeth	Wiesinger
Hans	Becker	Elsbeth	Wiesinger
Elsbeth	Wiesinger	Else	Mönike
Fritz	Hirsch	Else	Mönike
Werner	Reiter	Else	Mönike
Amelia	Kunze	Else	Mönike

4. Close the test results window, and leave **Framework Manager** open for the next demo.

Results:

You created a model query subject to function as an alias for the EMPLOYEE data source query subject, allowing you to resolve a recursive relationship in EMPLOYEE_HISTORY.

Business Analytics software

IBM

Modifying Relationships

- You can modify:
 - keys
 - cardinalities
 - operator
 - expression

Name: PRODUCT_TYPE_PRODUCT <-> SALES_TARGET

Query subject: Product Type & Product

Query subject: SALES_TARGET

Relationship impact: Each SALES_TARGET has one and only one Product Type & Product.
Each Product Type & Product has one or more SALES_TARGET.

Expression: Product Type & Product.PRODUCT_TYPE_CODE = SALES_TARGET.PRODUCT_TYPE_CODE

Relationship modifications are reflected in the generated SQL. For example, optional cardinality on one end creates a left outer join in the SQL, while optional cardinality on both ends creates a full outer join.

You can modify the operator to further define how the objects are related to one another. For example, to ensure that the related query subjects do not return data outside of a "posted date", you could define the following relationship:
FactTable.FactDate <= SecurityTable.PostedDate

You can also use the Expression editor to create complex join conditions. For example, you can extend the join syntax to include additional filter criteria. This lets you create a filter that is only applied when items from both query subjects are used, and leaves the individual query subjects unrestricted in other query scenarios.

Demo 4: Create a Complex Relationship Expression

Purpose:

There is a filter on the EMPLOYEE_HISTORY data source query subject that limits the data returned to current records for employees. However, Human Resources require both current and historical employee data. To support this, you will move the filter from the EMPLOYEE_HISTORY to the relationship expression between it and EMPLOYEE. You will then create a model query subject that acts as an alias to EMPLOYEE, and define a relationship between it and EMPLOYEE_HISTORY. Human resources staff can then query this alias to generate their reports.

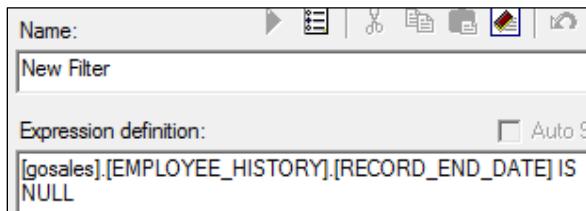
Component: Framework Manager

Project: GO Operational

Task 1. View the existing EMPLOYEE_HISTORY filter and move it to a relationship expression.

1. In Foundation Objects View > gosales, double-click **EMPLOYEE_HISTORY**, and then click the **Filters** tab.
2. In the **Source** column beside **New Filter**, click the **ellipsis**.

The results appear as follows:



This is a filter applied on a data source query subject to retrieve only current records for each employee (the one with no End Date assigned).

3. Copy the expression in the **Expression definition** pane, and then click **Cancel**.
4. With the filter selected, click **Delete**, and then click **OK**.
You will now test the effect of removing the filter.

5. Select and **Test** the following items:

Query Subject	Query Item
EMPLOYEE	EMPLOYEE_CODE FIRST_NAME LAST_NAME
EMPLOYEE_HISTORY	EMPLOYEE_HISTORY_CODE MANAGER

The results appear as follows:

Test results			
EMPLOYEE_CODE	FIRST_NAME	LAST_NAME	EMPLOYEE_HISTOR
10323	Aagdie	Heiman	30789
10004	Denis	Pagé	30000
10005	Élizabeth	Michel	30001
10006	Émile	Clemont	30002
10007	Étienne	Jauvin	30003
10012	Elsbeth	Wiesinger	30004
10013	Else	Mönike	30005
10014	Frank	Fuhlroth	30006
10015	Gunter	Erler	30007
10016	Björn	Winkler	30956
10016	Björn	Winkler	30008
10017	Fritz	Hirsch	30009
10018	Jörg	Kunze	30010
10019	Silvano	Allessori	30011
10020	Maria	Iacobucci	30938
10020	Maria	Iacobucci	30012

Some of the employees, such as 10016 and 10020, have more than one record. These are their current and historical records. This is the type of data that the human resources staff requires.

6. Click **Close**.
7. Right-click **EMPLOYEE_HISTORY**, click **Launch Context Explorer**, and then click **Show Related Objects**.
8. Double-click the relationship between **EMPLOYEE_HISTORY** and **EMPLOYEE**.

The Relationship Definition dialog opens.

9. In the bottom-right corner, click the **ellipsis** beside the **Expression** pane.
10. At the end of the expression, type **and**, and then paste the syntax from the filter.
11. Click **OK**.

The results appear as follows:

Name: EMPLOYEE <-> EMPLOYEE_HISTORY

Query subject: EMPLOYEE

Query subject: EMPLOYEE_HISTORY

Relationship impact: Each EMPLOYEE_HISTORY has one and only one EMPLOYEE. Each EMPLOYEE has one or more EMPLOYEE_HISTORY.

Expression: **EMPLOYEE.EMPLOYEE_CODE = EMPLOYEE_HISTORY.EMPLOYEE_CODE and EMPLOYEE_HISTORY.RECORD_END_DATE IS NULL**

The link between the two query subjects is no longer present since the expression contains non-join specific syntax.

12. Click the **Relationship SQL** tab, and then scroll down to the **where** clause.

The results appear as follows:

```

Sample SQL statement, using the relationship:
SELECT L1.EMPLOYEE_CODE AS L1_EMPLOYEE_CODE,
       L1.EMPLOYEE_HISTORY_PARENT AS L1_EMPLOYEE_HISTORY_PARENT,
       L1.EMPLOYEE_CODE AS L1_EMPLOYEE_CODE1,
       L1.RECORD_START_DATE AS RECORD_START_DATE,
       L1.RECORD_END_DATE AS RECORD_END_DATE,
       L1.POSITION_CODE AS POSITION_CODE,
       L1.POSITION_START_DATE AS POSITION_START_DATE,
       L1.MANAGER_CODE AS MANAGER_CODE,
       L1.MANAGER AS MANAGER,
       L1.MANAGER_MB AS MANAGER_MB,
       L1.BRANCH_CODE AS BRANCH_CODE,
       L1.ORGANIZATION_CODE AS ORGANIZATION_CODE
  FROM GOSALES..GOSALESHR.EMPLOYEE L1
 WHERE ((EMPLOYEE.EMPLOYEE_CODE = EMPLOYEE_HISTORY.EMPLOYEE_CODE)
        AND (EMPLOYEE_HISTORY.RECORD_END_DATE IS NULL))
  
```

The compound expression is reflected in the generated SQL.

13. Click the **Test** button.

Multiple records per employee (i.e. 10016 and 10020) no longer exist. This filter is applied when items from both query subjects, or the relationship between them, is used in a query. For example, if you query items from EMPLOYEE_HISTORY and SALES_TARGET_FACT, the EMPLOYEE query subject is in the query path, and therefore the filter will be applied.

14. Click **OK**, and then close the **Context Explorer**.

Task 2. Create an alias for EMPLOYEE using a model query subject.

Human Resources need to be able to report on employee data without having historical data restricted.

1. In the **Project Viewer**, right-click **EMPLOYEE**, and then click **Merge in New Query Subject**.
2. Click **No** to creating the underlying relationships.
3. Rename the **EMPLOYEE_EMPLOYEE** to **Employee (Human Resources)**.
4. Create a relationship between **Employee (Human Resources)** (1..1) and **EMPLOYEE_HISTORY** (1..n) on **EMPLOYEE_CODE**.
5. Click **OK**, and then click **No** when asked to create other underlying relationships.

6. Select and **Test** the following items:

Query Subject	Query Item
Employee (Human Resources)	EMPLOYEE_CODE FIRST_NAME LAST_NAME
EMPLOYEE_HISTORY	EMPLOYEE_HISTORY_CODE MANAGER

The results appear as follows:

Test results				
EMPLOYEE_CODE	FIRST_NAME	LAST_NAME	EMPLOYEE_HISTORY_CODE	MANAGER
10323	Aaghie	Heiman	30789	Amaury Moreau
10004	Denis	Pagé	30000	Dietz Krieger
10005	Élizabeth	Michel	30001	Frédéric Samson
10006	Émile	Clermont	30002	Frédéric Samson
10007	Étienne	Jauvin	30003	Frédéric Samson
10012	Elsbeth	Wiesinger	30004	Else Mörike
10013	Else	Mörike	30005	Barbara Samuelsen
10014	Frank	Fuhlroth	30006	Georges Saint-Germain
10015	Gunter	Erler	30007	Elsbeth Wiesinger
10016	Björn	Winkler	30956	Gretchen Goetschy
10016	Björn	Winkler	30008	Fritz Hirsch
10017	Fritz	Hirsch	30009	Else Mörike
10018	Jörg	Kunze	30010	Denis Pagé
10019	Silvano	Allessori	30011	Maria Iacobucci
10020	Maria	Iacobucci	30938	Derek Hirsch
10020	Maria	Iacobucci	30012	Fabian Tibor
10021

The historical records are now returned when querying between the employee alias query subject and EMPLOYEE_HISTORY and can be leveraged by the human resources staff.

7. Click **Close, save** and **Close** the project, and then close Framework Manager.

Results:

By moving a filter from the EMPLOYEE_HISTORY query subject into the relationship between EMPLOYEE_HISTORY and EMPLOYEE, you ensured that both query subjects need to be queried in order for the results to be filtered. You also created an alias for EMPLOYEE and configured its relationship with EMPLOYEE_HISTORY so that HR staff can use it to report on both current and historical employee data.

Summary

- You should now be able to:
 - understand the benefits of using model query subjects
 - use aliases to avoid ambiguous joins
 - merge query subjects to create as view behavior
 - resolve a recursive relationship
 - create a complex relationship expression

© 2015 IBM Corporation

This material is meant for IBM Academic Initiative use only. NOT FOR RESALE

© 2003, 2015, IBM Corporation

This guide contains proprietary information which is protected by copyright. No part of this document may be photocopied, reproduced, or translated into another language without a legal license agreement from IBM Corporation.



Modeling for Predictable Results: Consolidate Metadata

IBM Cognos BI



Business Analytics software

© 2015 IBM Corporation

This material is meant for IBM Academic Initiative use only. NOT FOR RESALE

© 2003, 2015, IBM Corporation

This guide contains proprietary information which is protected by copyright. No part of this document may be photocopied, reproduced, or translated into another language without a legal license agreement from IBM Corporation.

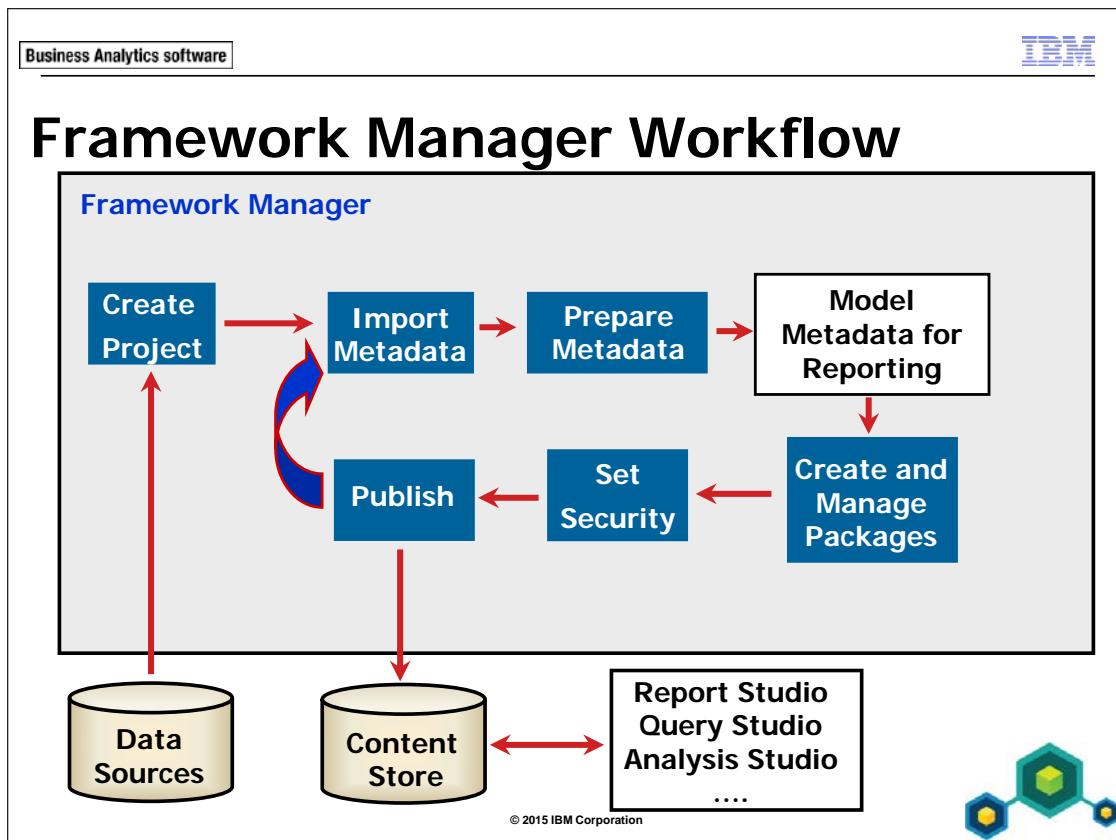
Objectives

- At the end of this module, you should be able to:
 - create virtual facts to simplify writing queries
 - create virtual dimensions to resolve fact-to-fact joins
 - create a consolidated modeling layer for presentation purposes
 - consolidate snowflake dimensions with model query subjects
 - simplify facts by hiding unnecessary codes

© 2015 IBM Corporation

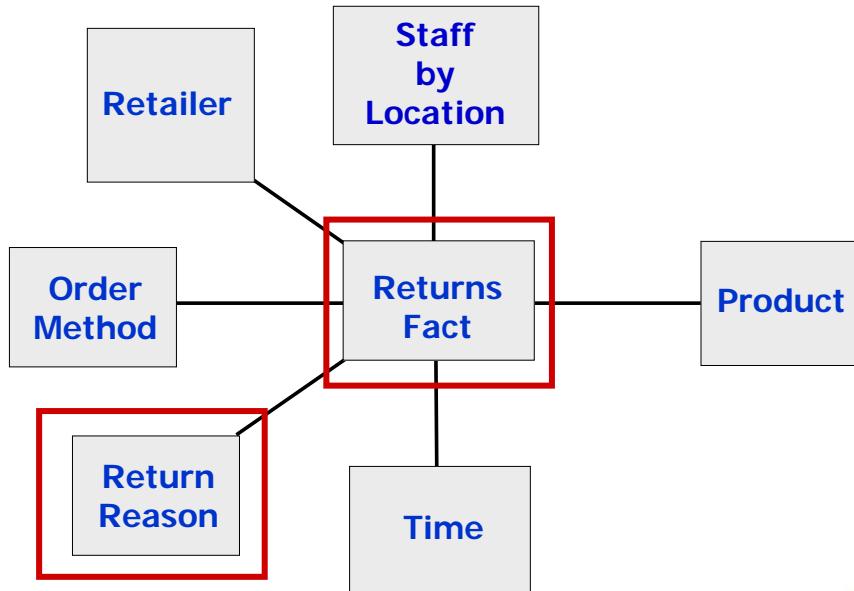
Unless otherwise specified in demo or workshop steps, you will always log on to IBM Cognos in Local LDAP namespace using the following credentials:

- User ID: admin
- Password: Education1



This module deals with creating a Consolidation View in which model query subjects are used to create a layer of consolidated metadata to be presented to authors. This layer will insulate reports from changes to the underlying data source.

Requirements Review: Returns



In the next demo, you will import Returns metadata and model it, so that the report author can create a report that examines Returns data for specific dimensions.

Business Analytics software

IBM

Examine Returns Data

- in the data source returns are related to order details
- this does not support reporting requirements, such as returns by staff, order method, and product

```
classDiagram ORDER_DETAILS "1..1" -- "0..n" RETURNED_ITEM
```

The diagram shows two UML class boxes. The left box is labeled 'ORDER_DETAILS' and the right box is labeled 'RETURNED_ITEM'. A line connects them with multiplicity '1..1' on the left and '0..n' on the right.

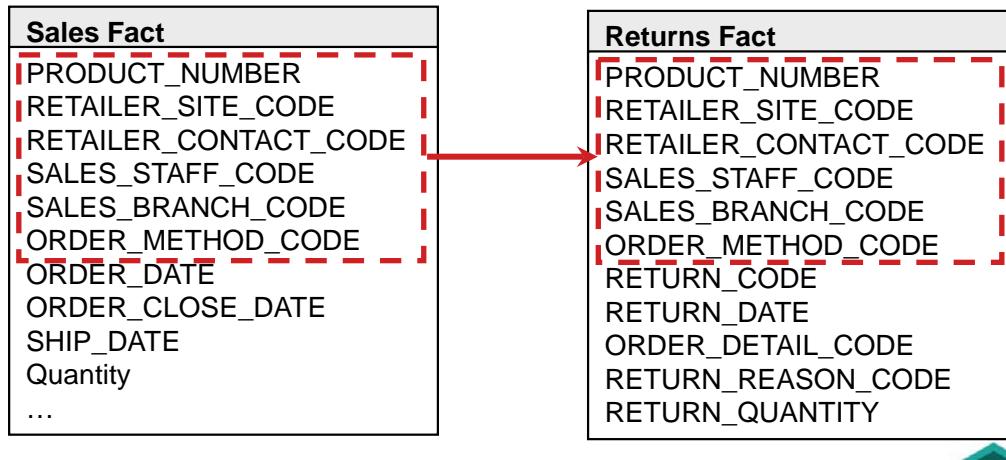
© 2015 IBM Corporation

The nature of the data does not currently let you directly report returns by staff, order method, or product.

You will need to use modeling techniques to make returns fact data appear directly related to the required dimensions.

Creating Virtual Facts

- the Returns Fact requires relationships with the same dimensions as Sales Fact



© 2015 IBM Corporation



Because Returns are so closely related to Sales, they should have the same relationships to other dimensions, such as Products, that Sales have. This gives authors the ability to write logical reports, such as "which products were returned", without having to include Sales information to get data for Returns.

You can add the missing context by creating a model query subject with the query items (keys) required to create relationships between Returns Fact and the required dimensions. The new model query subject behaves like a virtual fact table in a star schema data warehouse. You are simply creating a view that has the required items, and then creating relationships. Keys are hidden so that only facts are visible to authors.

Demo 1: Create a Virtual Fact

Purpose:

Report authors need to create reports on returns (such as how many returns are on a particular product). You will support this by creating a Returns Fact model query subject, and defining relationships between it and the related dimensional model query subjects. At the end of this demo, you will have modeled a virtual star schema.

Component: **Framework Manager**

Project: **GO Operational**

Task 1. Import metadata for Returns.

1. In **Framework Manager**, open the **GO Operational** model located at **C:\Edcognos\B5A52\CBIFM-Start Files\Module 8\GO Operational**. If prompted, log in as User ID **admin**, and Password **Education1**.
2. In the **Project Viewer** pane, right-click the **gosales** namespace, and then click **Run Metadata Wizard**.
3. Ensure that **Data Sources** is selected, and then click **Next**.
4. Ensure **GOSALES** is selected, and then click **Next**.
5. In the list of objects, expand **GOSALES > Tables**, and then select **RETURNED_ITEM** and **RETURN_REASON**.
6. Click **Next**.
7. On the **Generate Relationships** page, select **Both**.

Note: If you do not select Both, you will get a cross-join error when creating the Model Query Subject.

8. Click **Import**, and then click **Finish**.
9. Expand the two new data source query subjects to view their contents.
RETURNED_ITEM will be used to create Returns Fact and RETURN_REASON will be used as a dimension to provide context to returns.
You are now ready to create a new virtual fact by combining query items from several data source query subjects into one model query subject.

Task 2. Create the Returns Fact.

1. In the **gosales** namespace, create a new model query subject called **Returns Fact**, and then click **OK**.
2. In the **Query Subject Definition**, expand **Foundation Objects View > gosales**, and then add the following query items to the definition:

Query Subject	Query Item
RETURNED_ITEM	RETURN_CODE RETURN_DATE ORDER_DETAIL_CODE RETURN_REASON_CODE
ORDER_HEADER (in Original Sales Objects folder)	RETAILER_SITE_CODE SALES_STAFF_CODE SALES_BRANCH_CODE ORDER_METHOD_CODE
ORDER_DETAILS (in Original Sales Objects folder)	PRODUCT_NUMBER
RETURNED_ITEM	RETURN_QUANTITY

Note: You could also move the three RETURNED_ITEM query items into Sales Fact (instead of creating Returns Fact) providing you also add a determinant to Sales Fact to handle multiple levels of granularity. However, doing so would cause all authored reports and queries on Sales Fact to perform an outer join to RETURNED_ITEM. Since most Sales Fact queries are not about returns, this would be a lot of unnecessary processing.

3. Click **OK**, expand **Returns Fact**, and then rename **RETURN_QUANTITY** to **Return Quantity**.
4. Drag **Returns Fact** to just below **SALES_TARGET**, and then drag **RETURN_REASON** below **RETAILER_TYPE**.
You now have three fact query subjects (Sales Fact, SALES_TARGET, and Returns Fact), followed by the dimension query subjects.
5. Drag **RETURNED_ITEM** to the **Original Sales Objects** folder, and then rename the folder to **Original Sales & Returns Objects**.

Task 3. Create relationships between Returns Fact and existing model query subjects.

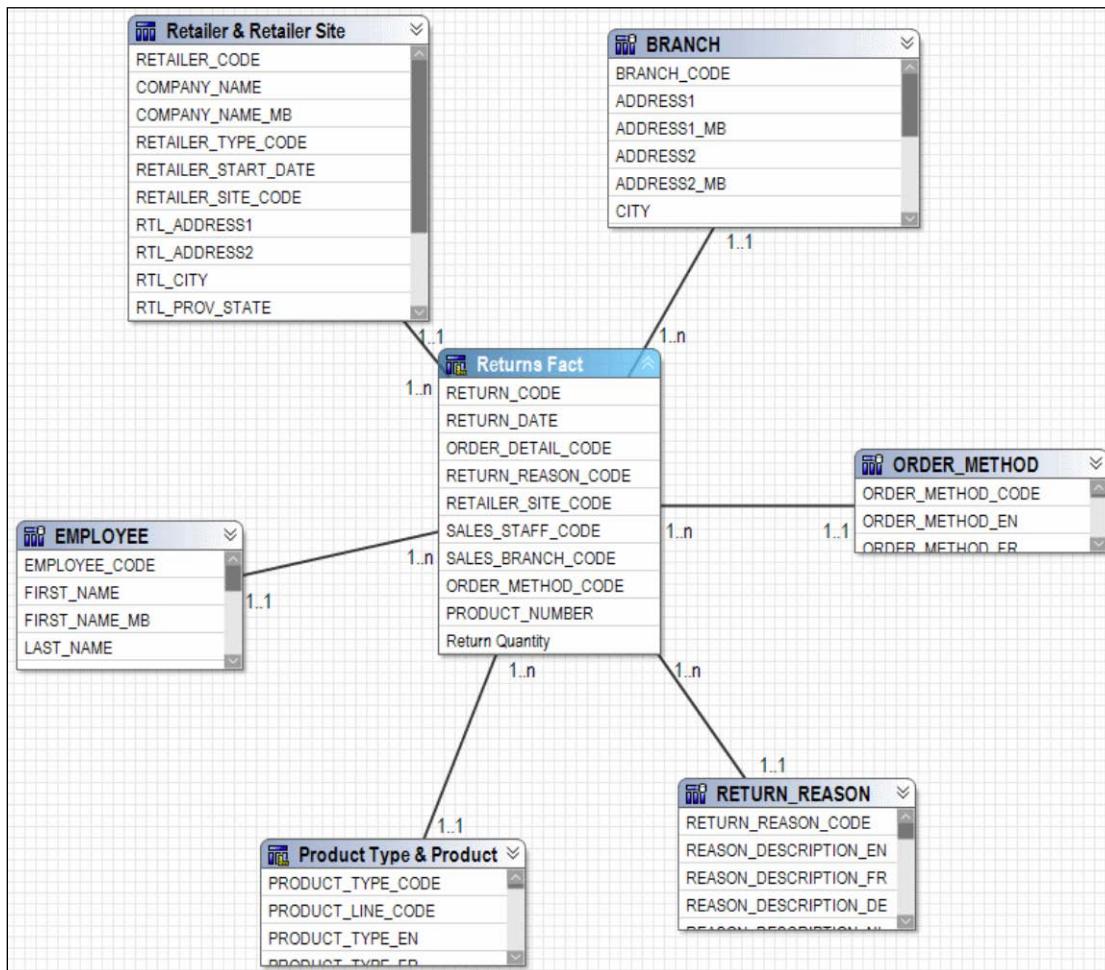
1. Create the following relationships, clicking **No** if asked to replicate the existing underlying relationships:

- **RETURN_REASON** (RETURN_REASON_CODE, 1..1) to **Returns Fact** (RETURN_REASON_CODE, 1..n)
- **ORDER_METHOD** (ORDER_METHOD_CODE, 1..1) to **Returns Fact** (ORDER_METHOD_CODE, 1..n)
- **Product Type & Product** (PRODUCT_NUMBER, 1..1) to **Returns Fact** (PRODUCT_NUMBER, 1..n)
- **EMPLOYEE** (EMPLOYEE_CODE, 1..1) to **Returns Fact** (SALES_STAFF_CODE, 1..n)
- **Retailer and Retailer Site** (RETAILER_SITE_CODE, 1..1) to **Returns Fact** (RETAILER_SITE_CODE, 1..n)
- **BRANCH** (BRANCH_CODE, 1..1) to **Returns Fact** (SALES_BRANCH_CODE, 1..n)

Note: These are the same relationships that Sales Fact has (with the exception of the one to RETURN_REASON). You could argue that these relationships should be 0..n instead of 1..n. For example, you may wish to report on the products, staff, or retailers with no returns. The scope of our reporting package, however, does not include those situations. You can change the relationships to 0..n if the business case requires it, but if it doesn't, you should keep them at 1..n for the better performance.

- Right-click **Returns Fact**, click **Launch Context Explorer**, and then click **Show Related Objects**.

Arrange the objects as follows:



Notice that Returns Fact appears in a star schema in which the fact query subject is surrounded by its related dimensions.

- Close **Context Explorer**.

Task 4. Delete a redundant relationship.

The original RETURNED_ITEM has a relationship to RETURN_REASON. This relationship is no longer required since the new Returns Fact model query subject now has a relationship to RETURN_REASON.

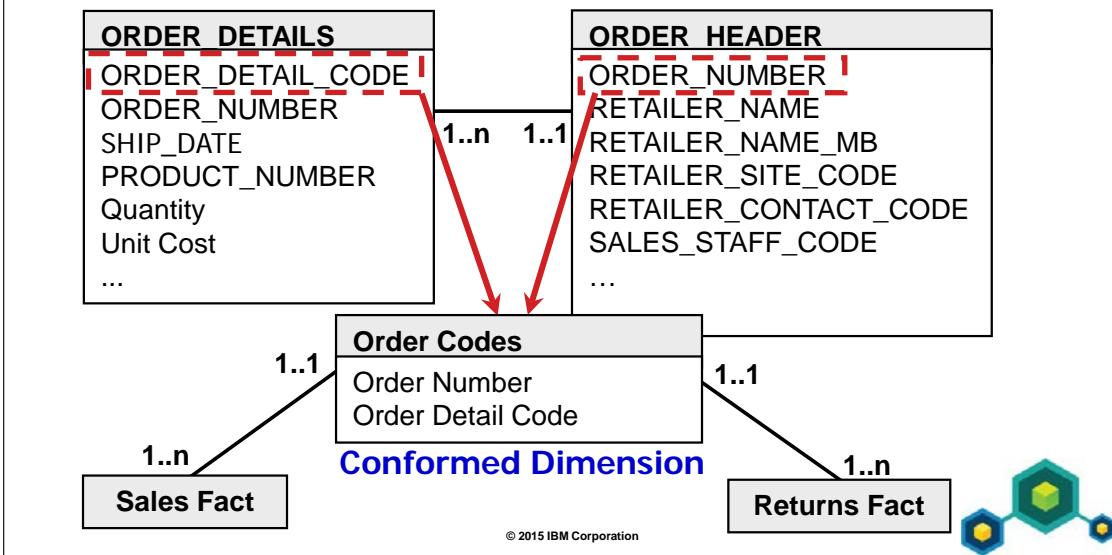
1. Right-click **RETURN_REASON**, click **Launch Context Explorer**, and then click **Show Related Objects**.
2. Delete the relationship between **RETURN_REASON** and **RETURN_ITEM**.
3. Close **Context Explorer**, **Save** the project, and leave Framework Manager open for the workshop.

Results:

You created the Returns Fact, as well as the relationships that are required with existing model query subjects. This virtual star schema supports queries on returns, such as how many returns there have been for a particular product.

Creating Virtual Dimensions

- Add dimensional query items from ORDER_HEADER and ORDER_DETAILS to create Order Codes.



Certain non-fact query items from fact tables can be placed in a model query subject to ensure that they are viewed as coming from a dimension by IBM Cognos.

For example, ORDER_NUMBER and ORDER_DETAIL_CODE are required for reporting purposes and are located in ORDER_HEADER and ORDER_DETAILS respectively. By placing these query items in a new model query subject called Order Codes, you create a query subject that acts like a virtual dimension table as seen in a star schema data warehouse. This ensures that the dimensional query items are not missing from the final model presentation.

In the underlying data source, the only way to query returns is through ORDER_DETAILS, which is a fact-to-fact join. Once created, the Order Codes query subject can act as a conformed dimension between Sales Fact and Returns Fact.

Workshop 1: Create a Virtual Dimension

Analyzing the model design for flexibility, you see two reasons to create an Order Codes dimensional model query subject. First, it creates a virtual dimension that ensures that dimensional query items in ORDER_DETAILS and ORDER_HEADER are available in a dimension. Second, it allows report authors to stitch together queries on Sales Fact and Returns Fact in a single report, by using it as a conformed dimension.

Create a virtual dimension as follows:

- In gosales, create an **Order Codes** model query subject containing:
 - **ORDER_DETAILS.ORDER_DETAIL_CODE**
 - **ORDER_HEADER.ORDER_NUMBER**
- Create relationships from **Order Codes** to both **Sales Fact** and **Returns Fact**, based on the **ORDER_DETAIL_CODE**.
- Test the use of **Order Codes** as a conformed dimension:
 - test **Sales Fact.ORDER_NUMBER**, **Sales Fact.Quantity**, and **Returns Fact.Return Quantity**
 - test **Order Codes.Order Number**, **Sales Fact.Quantity**, and **Returns Fact.Return Quantity**.
 - compare the SQL between the two tests

For more information about where to work and the workshop results, refer to the Tasks and Results section that follows. If you need more information to complete a task, refer to earlier demos for detailed steps.

Workshop 1: Tasks and Results

Task 1. Create a new model query subject.

- In the **gosales** namespace, create a model query subject named **Order Codes**.
- From the **Original Sales & Returns Objects** folder, add the following query items:
 - **ORDER_HEADER.ORDER_NUMBER**
 - **ORDER_DETAILS.ORDER_DETAIL_CODE**
- Rename the query items to **Order Number** and **Order Detail Code**.
- Drag **Order Codes** to below **Manager** to maintain the alphabetical order.

Task 2. Create relationships for Order Codes.

- Create the following relationships with **Order Codes** (do not replicate existing underlying relationships):
 - **Order Codes** (Order Detail Code, 1..1) to **Sales Fact** (ORDER_DETAIL_CODE, 1..n)
 - **Order Codes** (Order Detail Code, 1..1) to **Returns Fact** (ORDER_DETAIL_CODE, 1..n)

Task 3. Test Order Codes as a conformed dimension.

- Test the following items (with Auto Sum):
 - Sales Fact. ORDER_NUMBER
 - Sales Fact. Quantity
 - Returns Fact. Return Quantity
- Click the **Query Information** tab.

The results appear as follows:

```
Cognos SQL
select
    D2.ORDER_NUMBER as ORDER_NUMBER,
    D2.Quantity as Quantity,
    D3.Return_Quantity as Return_Quantity
from
    (select
        Sales_Fact.ORDER_NUMBER as ORDER_NUMBER,
        XSUM(Sales_Fact.Quantity for Sales_Fact.ORDER_NUMBER) as Quantity
    from
        (select
            ORDER_DETAILS.ORDER_NUMBER as ORDER_NUMBER,
            ORDER_DETAILS.QUANTITY as Quantity
        from
            GOSALES..GOSALES.ORDER_DETAILS ORDER_DETAILS,
            GOSALES..GOSALES.ORDER_HEADER ORDER_HEADER
        where
            (ORDER_HEADER.ORDER_NUMBER = ORDER_DETAILS.ORDER_NUMBER)
        ) Sales_Fact
    group by
        Sales_Fact.ORDER_NUMBER
    ) D2,
    (select distinct
        XSUM(Returns_Fact.Return_Quantity) as Return_Quantity
    from
        (select
            RETURNED_ITEM.RETURN_QUANTITY as Return_Quantity
        from
            GOSALES..GOSALES.RETURNED_ITEM RETURNED_ITEM,
            GOSALES..GOSALES.ORDER_HEADER ORDER_HEADER,
            GOSALES..GOSALES.ORDER_DETAILS ORDER_DETAILS
        where
            (ORDER_DETAILS.ORDER_DETAIL_CODE = RETURNED_ITEM.ORDER_DETAIL_CODE) and
            (ORDER_HEADER.ORDER_NUMBER = ORDER_DETAILS.ORDER_NUMBER)
        ) Returns_Fact
    ) D3
```

No stitch query (full outer-join) is performed, and Return Quantity returns only one value for each record.

- Test the following items (with Auto Sum):
 - Order Codes. Order Number
 - Sales Fact. Quantity
 - Returns Fact. Return Quantity

This material is meant for IBM Academic Initiative use only. NOT FOR RESALE

- Click the **Query Information** tab.

The results appear as follows:

```
Cognos SQL
select
    coalesce(D2.Order_Number,D3.Order_Number) as Order_Number,
    D2.Quantity as Quantity,
    D3.Return_Quantity as Return_Quantity
from
    (select
        Order_Codes.Order_Number as Order_Number,
        XSUM(Sales_Fact.Quantity for Order_Codes.Order_Number) as Quantity
    from
        (select
            ORDER_DETAILS.ORDER_NUMBER as Order_Number,
            ORDER_DETAILS.ORDER_DETAIL_CODE as Order_Detail_Code
        from
            GOSALES..GOSALES.ORDER_DETAILS ORDER_DETAILS
        ) Order_Codes,
        (select
            ORDER_DETAILS.ORDER_DETAIL_CODE as ORDER_DETAIL_CODE,
            ORDER_DETAILS.QUANTITY as Quantity
        from
            GOSALES..GOSALES.ORDER_DETAILS ORDER_DETAILS,
            GOSALES..GOSALES.ORDER_HEADER ORDER_HEADER
        where
            (ORDER_HEADER.ORDER_NUMBER = ORDER_DETAILS.ORDER_NUMBER)
        ) Sales_Fact
    where
        (Order_Codes.Order_Detail_Code = Sales_Fact.ORDER_DETAIL_CODE)
    group by
        Order_Codes.Order_Number
    ) D2
    full outer join
    (select
        Order_Codes.Order_Number as Order_Number,
        XSUM(Returns_Fact.Return_Quantity for Order_Codes.Order_Number) as Return_Quantity
    from
        (select
            ORDER_DETAILS.ORDER_NUMBER as Order_Number,
            ORDER_DETAILS.ORDER_DETAIL_CODE as Order_Detail_Code
        from
            GOSALES..GOSALES.ORDER_DETAILS ORDER_DETAILS
        ) Order_Codes,
        (select
```

The correct data is returned, and the SQL shows that a stitch query (full outer join) merged the two fact result sets together.

- Leave Framework Manager open for the next demo.

Business Analytics software

IBM

Consolidating Metadata

- Foundation Objects View:
 - holds all the foundation query subjects and their relationships
- Consolidation View:
 - uses model query subjects to consolidate foundation metadata for presentation
 - contains filters and calculations where required

```

graph TD
    A[Presentation View] --> B[Consolidation View]
    B --> C[Foundation Objects View]
  
```

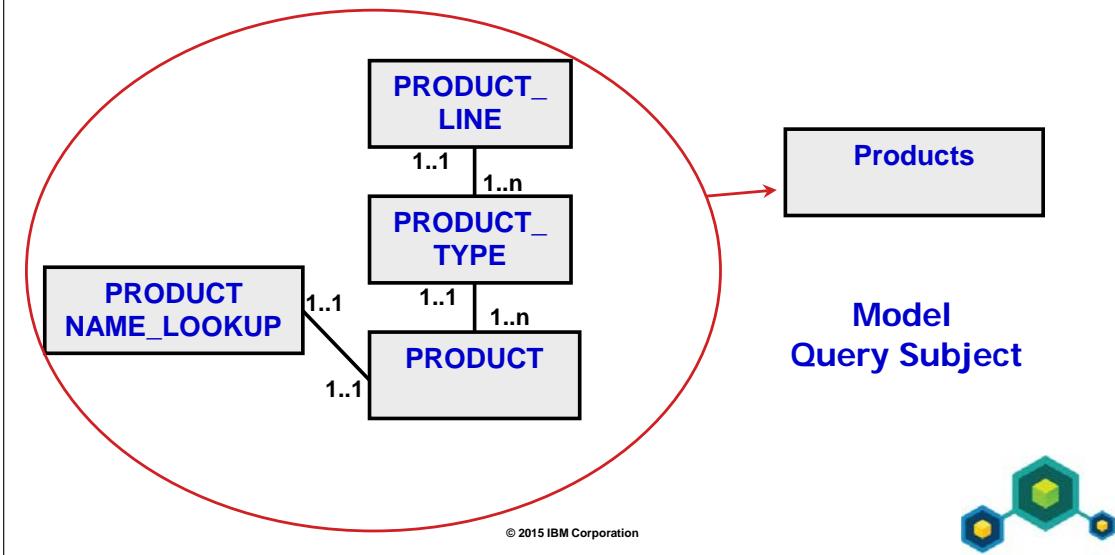
© 2015 IBM Corporation

The Consolidation View provides a layer of metadata that is separate from the foundation layer that contains all the relationships. To understand query paths, view the Foundation Objects View. To view how metadata will be presented to authors, view objects in the Consolidation View.

Consolidating the metadata involves tasks such as consolidating snowflake dimensions into one model query subject, or hiding codes found in fact query subjects, which are used for relationships.

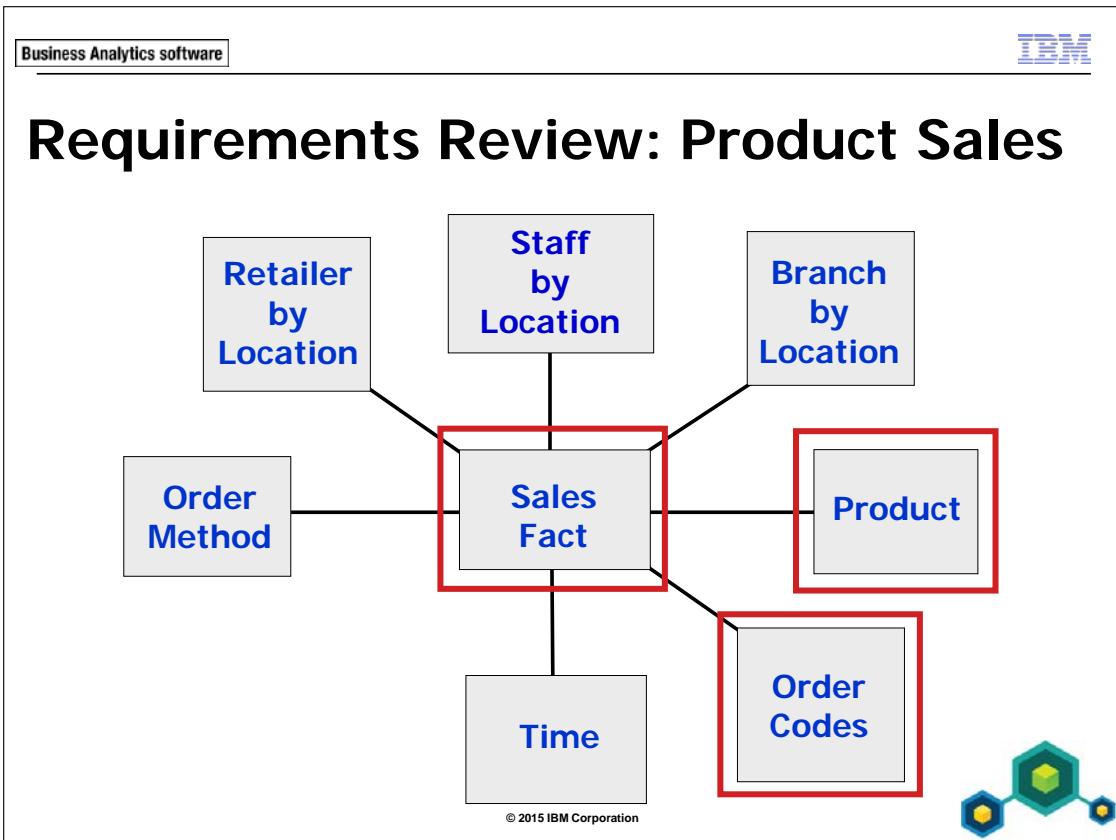
Consolidation Example

- Consolidate a snowflake dimension into one model query subject for presentation



An example of consolidating snowflake dimensions is placing query items from PRODUCT_LINE, PRODUCT_TYPE, PRODUCT, and PRODUCT_NAME_LOOKUP into a model query subject called Products.

You can also use the Consolidation View to neatly present your facts. For example you can create a fact query subject that only contains measures. The codes used in relationships found in the underlying Foundation Objects View are not visible to report authors.



In the next demo, you will consolidate product information into one model query subject in order to create a simplified Product dimension for report authors.

You will also create a simplified model query subject for Sales Fact. The new conformed dimension you created in the last workshop called Order Codes is shared between Sales Fact and Returns Fact.

Demo 2: Create the Consolidation View

Purpose:

To simplify the metadata view for authors, you will consolidate related query items into single query subjects. Authors would like access to codes but to reduce clutter you will organize them into query item folders.

Components: Framework Manager, IBM Cognos Workspace Advanced

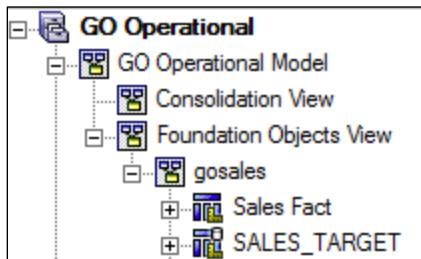
Project: GO Operational

Package: Consolidation View Test

Task 1. Create a new namespace for consolidated objects.

1. In the **Project Viewer** pane, in the **GO Operational Model** namespace, create a namespace called **Consolidation View**.
2. Drag the new namespace above **Foundation Objects View**.

The results appear as follows:



Task 2. Consolidate a snowflake dimension.

1. In the **Consolidation View** namespace, create a new model query subject called **Products**, and then click **OK**.
2. In the **Query Subject Definition**, expand **Foundation Objects View > gosales**, and then add the following query items to the definition:

Query Subject	Query Item
PRODUCT_LINE	PRODUCT_LINE_CODE PRODUCT_LINE_EN
Product Type & Product	PRODUCT_TYPE_CODE PRODUCT_TYPE_EN
PRODUCT_NAME_LOOKUP	PRODUCT_NAME PRODUCT_DESCRIPTION
Product Type & Product	PRODUCT_NUMBER PRODUCT_IMAGE INTRODUCTION_DATE DISCONTINUED_DATE

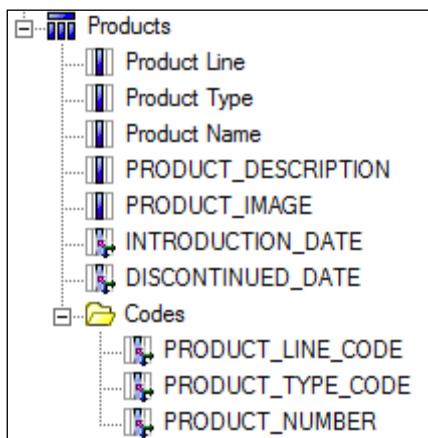
Note: You can also create this model query subject by selecting the individual query subjects (or the query items within them) and then merging them into a new model query subject.

3. Click **OK**, right-click **Products**, point to **Create**, and then click **Query Item Folder**.
4. Rename the folder to **Codes**.
5. Drag the following items into the **Codes** folder:
 - **PRODUCT_LINE_CODE**
 - **PRODUCT_TYPE_CODE**
 - **PRODUCT_NUMBER**

Authors still need these query items for activities like filtering and drill-through scenarios, so moving them into a subfolder is a good way to centralize such codes. It also organizes the metadata and makes the main dimensional query items easier to locate.

6. In the **Products** query subject, rename the first three query items as follows:
 - **Product Line**
 - **Product Type**
 - **Product Name**

The results appear as follows:



You would normally rename all items, but in order to save time items will be renamed for you in the starting point project at the beginning of the next module.

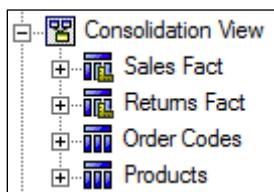
Task 3. Create simplified query subjects for presentation.

1. In the **Consolidation View** namespace, create a new model query subject called **Sales Fact** containing the following query items from **Sales Fact** in the **Foundation Objects View**:
 - **Quantity**
 - **Unit Cost**
 - **Unit Price**
 - **Unit Sale Price**
2. Create a new model query subject called **Returns Fact** containing the following query item from **Returns Fact** in the **Foundation Objects View**:
 - **Return Quantity**

3. Create a new model query subject called **Order Codes** containing the following query items from **Order Codes** in the **Foundation Objects View**:
 - **Order Number**
 - **Order Detail Code**

You create a new model query subject for Order Codes instead of simply creating a shortcut to the Order Codes in Foundation Objects View. This is partly to give us a consistent view in this layer, but it is also because shortcuts cannot be used to create star schema groupings, which you will later need to do to populate the Presentation View.

4. In the **Consolidation View** namespace, drag **Products** below **Order Codes**.
The results appear as follows:



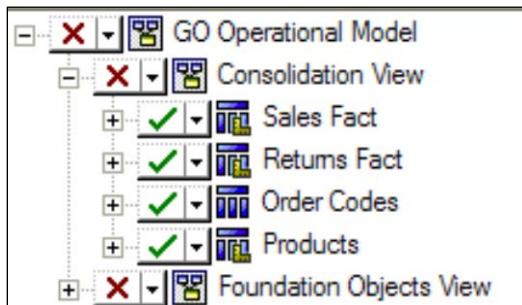
5. **Save** the project.

You now have a consolidated view of the data that authors can report from without having to know the underlying relationships or table complexity.

Task 4. Publish a package of the Consolidation View.

1. Right-click **Packages**, point to **Create**, and then click **Package**.
2. Name the package **Consolidation View Test**, and then click **Next**.
3. Clear **GO Operational Model**, expand **Consolidation View**, and select the four model query subjects.

The results appear as follows:



4. Click **Finish**, and then click **Yes** to open the **Publish Package** wizard.
5. Clear **Enable Model Versioning**, and then click **Next** twice.
6. Clear the **Verify the package** check box, click **Publish**, and then click **Finish**.

This material is meant for IBM Academic Initiative use only. NOT FOR RESALE

Task 5. Test the Package.

1. In **IBM Cognos Connection**, launch **IBM Cognos Workspace Advanced**.
2. Select the **Consolidation View Test** package, and create a new **List** report.
Note that only the selected query subjects appear in the data tree.
3. Create a query with the following query items:

Query Subject	Query Item
Order Codes	Order Number
Products	Product Line
Sales Fact	Quantity
Returns Fact	Return Quantity

Note. Product Name does not yet have a filter that only retrieves the local session's language. Therefore adding it to the report would return unexpected results.

The results appear as follows:

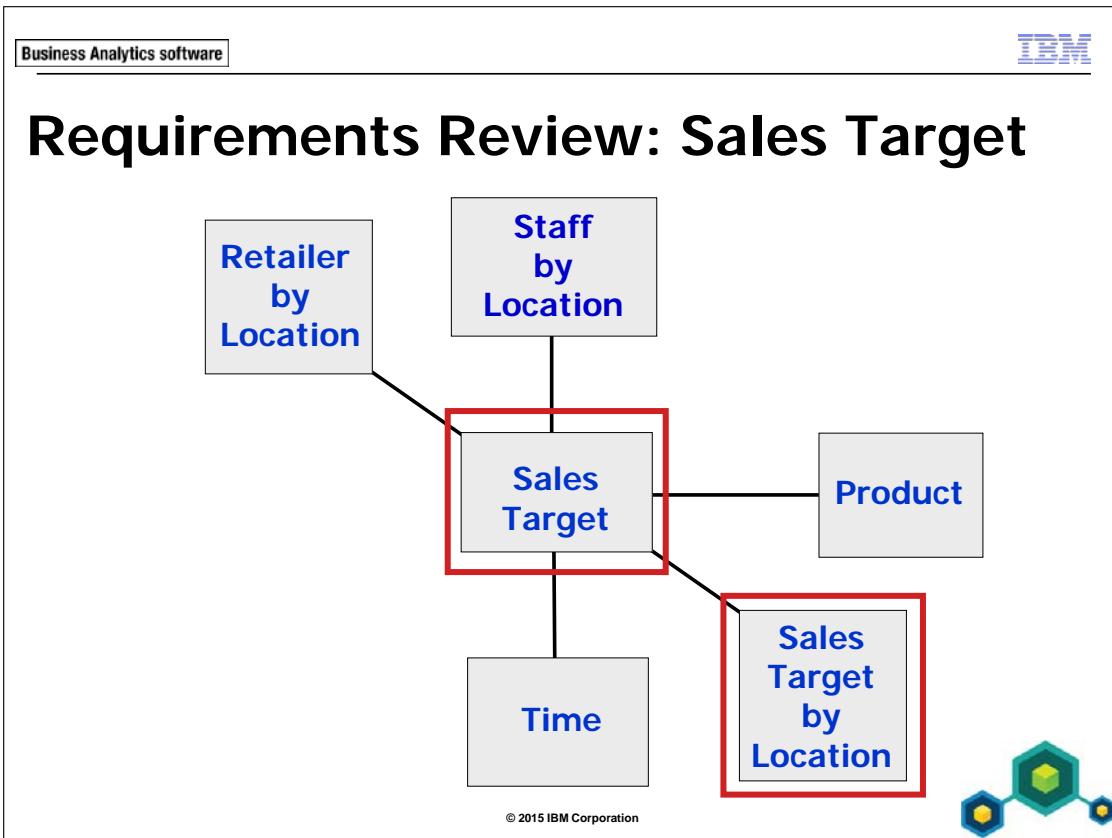
Order Number	Product Line	Quantity	Return Quantity
100001	Camping Equipment	256	
100001	Personal Accessories	92	
100002	Outdoor Protection	422	
100002	Personal Accessories	498	
100003	Outdoor Protection	4,359	
100003	Personal Accessories	107	19
100004	Camping Equipment	1,033	
100005	Golf Equipment	26	
100005	Personal Accessories	635	
100006	Outdoor Protection	7,774	

You have successfully created a report from a simplified and author-friendly model that hides the underlying complexity of the metadata.

4. Close the browser without saving the query, and leave Framework Manager open for the workshop.

Results:

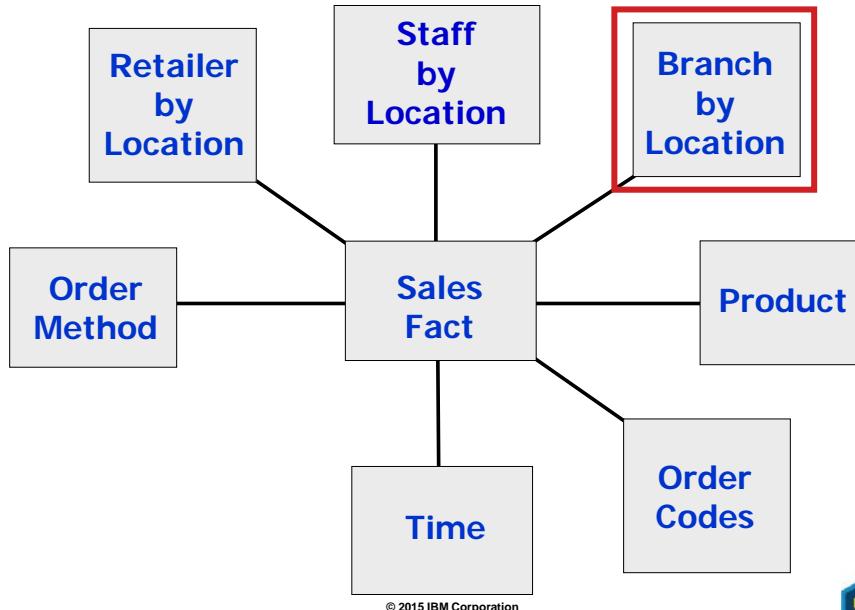
You have created consolidated and simplified query subjects to meet reporting needs, and you have tested the resulting package in IBM Cognos Workspace Advanced.



During the modeling process, you discovered another important dimension that can be used to report against Sales Target. In the next workshop, you will create a consolidated model query subject to allow users to report on sales targets by location.

You will also create a simplified view of Sales Target.

Requirements Review: Branch by Location



During the modeling process, you also discovered an alternate and important dimension that can be used to report against Sales Fact. In the next workshop, you will create a consolidated model query subject called Branch by Location.

Workshop 2: Consolidate and Simplify the Model for Presentation

You will continue to consolidate related query items into single query subjects to simplify the presentation of metadata for authors.

Create consolidated and simplified query subjects in the Consolidation View to meet these needs, as follows:

- Create a consolidated **Sales Target by Location** dimension from **SALES_REGION** and **COUNTRY**.
- Create a consolidated **Branch by Location** dimension from **Branch Region (alias)**, **Branch Country (alias)**, and **BRANCH**.
- Create a simplified **Sales Target Fact** containing **gosales.SALES_TARGET.>SALES_TARGET** and rename it to mixed case.
- Create a consolidated **Staff by Location**, containing:

Query Items and Calculations:	
Name	Source
Staff Region	Staff Region (alias).Staff Region
Staff Country	Staff Country (alias).Staff Country
Staff Address 1	Staff Branch (alias).Staff Address 1
Staff Address 2	Staff Branch (alias).Staff Address 2
Staff City	Staff Branch (alias).Staff City
Staff Prov/State	Staff Branch (alias).Staff Prov/State
Staff Postal Zone	Staff Branch (alias).Staff Postal Zone
First Name	EMPLOYEE.FIRST_NAME
Last Name	EMPLOYEE.LAST_NAME
Work Phone	EMPLOYEE.WORK_PHONE
Extension	EMPLOYEE.EXTENSION
Fax	EMPLOYEE.FAX
Email	EMPLOYEE.EMAIL
Manager	Manager.Manager First Name '' Manager.Manager Last Name
Position	POSITION_LOOKUP.POSITION_EN
Staff Region Code	Staff Region (alias).Staff Region Code
Staff Country Code	Staff Country (alias).Staff Country Code
Staff Branch Code	Staff Branch (alias).Staff Branch Code
Sales Staff Code	EMPLOYEE.EMPLOYEE_CODE
Manager Code	EMPLOYEE_HISTORY.MANAGER_CODE
Position Code	POSITION_LOOKUP.POSITION_CODE
Manager Work Phone	Manager.Manager Work Phone
Manager Extension	Manager.Manager Extension
Manager Fax	Manager.Manager Fax
Manager Email	Manager.Manager Email
Manager Position	Manager.Manager Position

This material is meant for IBM Academic Initiative use only. NOT FOR RESALE

- To display the full manager name, concatenate the manager's first and last name by changing the **Staff by Location > Manager** query item expression with the following.

```
[gosales].[Manager].[Manager First Name] || ' ' ||  
[gosales].[Manager].[Manager Last Name]
```

Save the project when you are finished.

For more information about where to work and the workshop results, refer to the Tasks and Results section that follows. If you need more information to complete a task, refer to earlier demos for detailed steps.

Workshop 2: Tasks and Results

Task 1. Create the consolidated Sales Target by Location query subject.

- In the **Consolidation View**, create the **Sales Target by Location** model query subject, containing the following items from the **Foundation Object View**:

Query Subject	Query Item
SALES_REGION	SALES_REGION_CODE SALES_REGION_EN
COUNTRY	COUNTRY_CODE COUNTRY_EN

- Rename the query items as follows:
 - Sales Target Region Code**
 - Sales Target Region**
 - Sales Target Country Code**
 - Sales Target Country**
- In **Sales Target by Location**, create a new query item folder named **Codes**, and move **Sales Target Region Code** and **Sales Target Country Code** into it.

Task 2. Create the consolidated Branch by Location query subject.

- In the **Consolidation View**, create the **Branch by Location** model query subject, containing the following items:

Query Subject	Query Item
Branch Region (alias)	Branch Region Code Branch Region
Branch Country (alias)	Branch Country Code Branch Country
BRANCH	BRANCH_CODE ADDRESS1 ADDRESS2 CITY PROV_STATE POSTAL_ZONE

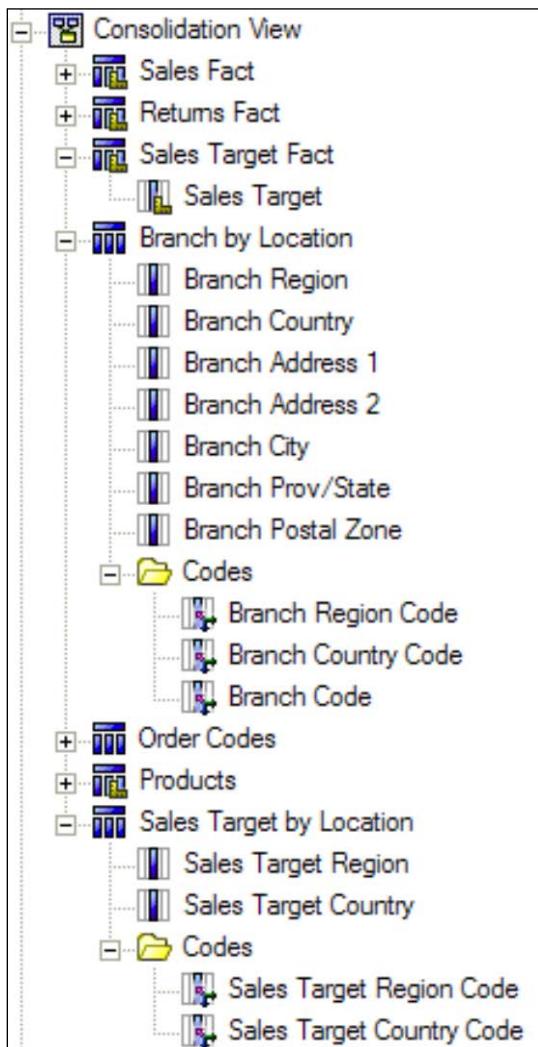
- In **Branch by Location**, create a new **Codes** query item folder, and move **Branch Region Code**, **Branch Country Code**, and **BRANCH_CODE** into it.
- Rename query items from **BRANCH** to mixed case with a **Branch** prefix (for example, **Branch Address1**).
- Drag **Branch by Location** below the **Returns Fact** query subject.

Task 3. Create a simplified Sales Target Fact.

- In the **Consolidation View**, create the **Sales Target Fact** model query subject, containing **SALES_TARGET.SALES_TARGET**.
- Rename the query item to **Sales Target**.

- Drag **Sales Target Fact** below **Returns Fact**.

The results appear as follows:



Task 4. Create the consolidated Staff by Location query subject.

- In the **Consolidation View**, create the **Staff by Location** model query subject, containing the following items, and the modify the **Name** to mixed case as seen below:

Query Items and Calculations:	
Name	Source
Staff Region	Staff Region (alias).Staff Region
Staff Country	Staff Country (alias).Staff Country
Staff Address 1	Staff Branch (alias).Staff Address 1
Staff Address 2	Staff Branch (alias).Staff Address 2
Staff City	Staff Branch (alias).Staff City
Staff Prov/State	Staff Branch (alias).Staff Prov/State
Staff Postal Zone	Staff Branch (alias).Staff Postal Zone
First Name	EMPLOYEE.FIRST_NAME
Last Name	EMPLOYEE.LAST_NAME
Work Phone	EMPLOYEE.WORK_PHONE
Extension	EMPLOYEE.EXTENSION
Fax	EMPLOYEE.FAX
Email	EMPLOYEE.EMAIL
Manager	EMPLOYEE_HISTORY.MANAGER
Position	POSITION_LOOKUP.POSITION_EN
Staff Region Code	Staff Region (alias).Staff Region Code
Staff Country Code	Staff Country (alias).Staff Country Code
Staff Branch Code	Staff Branch (alias).Staff Branch Code
Sales Staff Code	EMPLOYEE.EMPLOYEE_CODE
Manager Code	EMPLOYEE_HISTORY.MANAGER_CODE
Position Code	POSITION_LOOKUP.POSITION_CODE

- In **Staff by Location**, double-click the **Manager** query item, and replace the existing expression with the following:

[gosales].[Manager].[Manager First Name] || ' ' ||
 [gosales].[Manager].[Manager Last Name]

The results appear similar to the following:

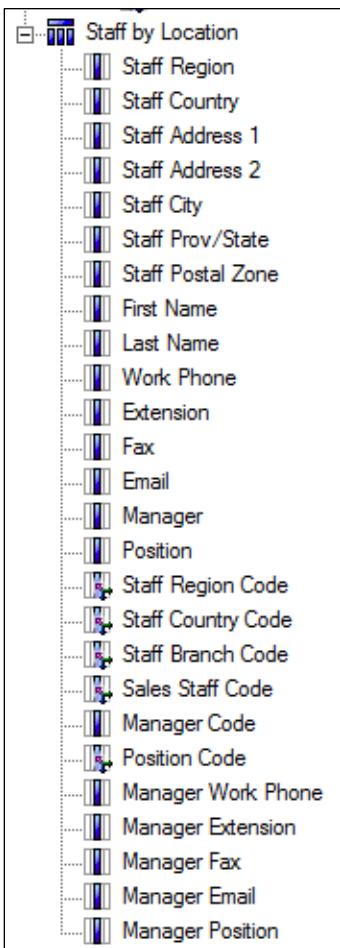
Test Results	
	Manager
	Chiyo Suzuki
	Denis Pagé
	Élizabeth Michel
	Émile Clermont
	Étienne Jauvin
	Elsbeth Wiesinger
	Elspeth

This material is meant for IBM Academic Initiative use only. NOT FOR RESALE

© 2003, 2015, IBM Corporation

This guide contains proprietary information which is protected by copyright. No part of this document may be photocopied, reproduced, or translated into another language without a legal license agreement from IBM Corporation.

- In the **Project Viewer**, double-click **Staff by Location**, and from **Foundation Objects View > gosales > Manager**, add the following items to the **Query Items and Calculations** pane:
 - **Manager Work Phone**
 - **Manager Extension**
 - **Manager Fax**
 - **Manager Email**
 - **Manager Position**
- Arrange the query items as shown below:



- **Save and Close** Framework Manager.

You consolidated and simplified query subjects in the Consolidation View to simplify the metadata for report authors.

Summary

- You should now be able to:
 - create virtual facts to simplify writing queries
 - create virtual dimensions to resolve fact-to-fact joins
 - create a consolidated modeling layer for presentation purposes
 - consolidate snowflake dimensions with model query subjects
 - simplify facts by hiding unnecessary codes

© 2015 IBM Corporation

This material is meant for IBM Academic Initiative use only. NOT FOR RESALE

© 2003, 2015, IBM Corporation

This guide contains proprietary information which is protected by copyright. No part of this document may be photocopied, reproduced, or translated into another language without a legal license agreement from IBM Corporation.

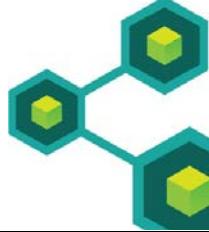


Calculations and Filters

IBM Cognos BI

Business Analytics software

© 2015 IBM Corporation



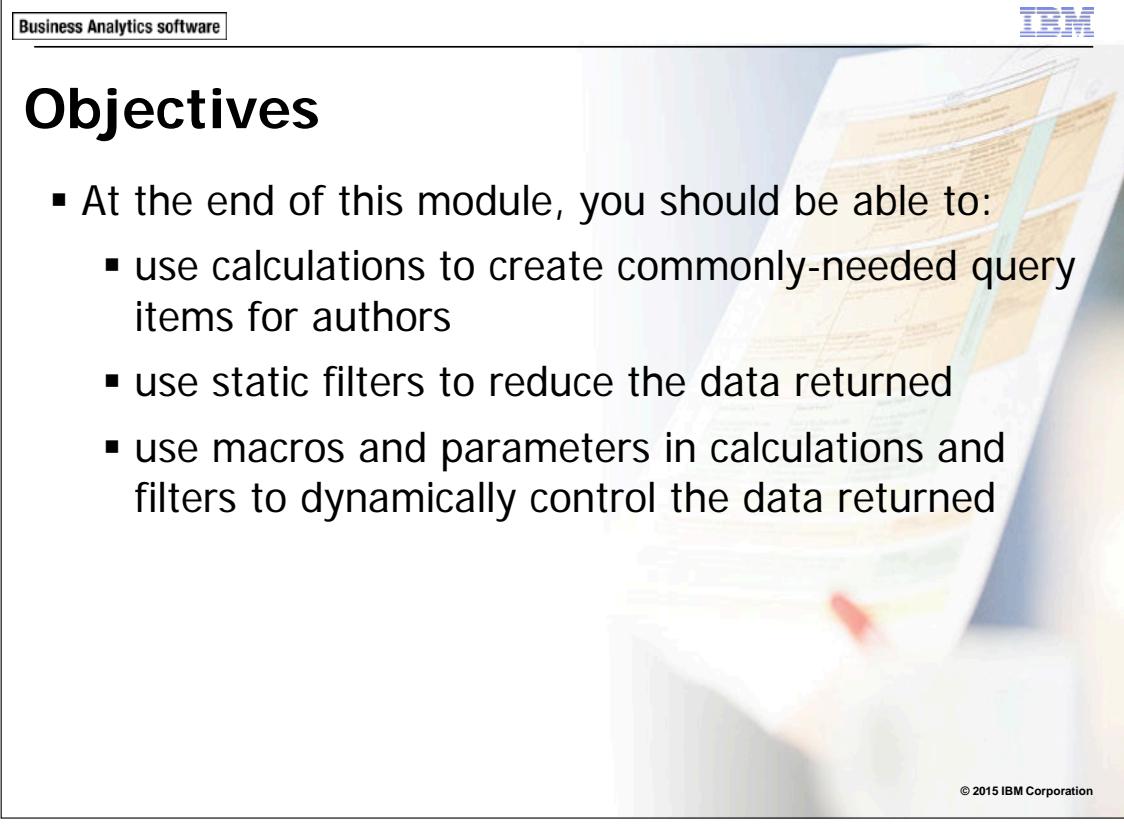
This material is meant for IBM Academic Initiative use only. NOT FOR RESALE

© 2003, 2015, IBM Corporation

This guide contains proprietary information which is protected by copyright. No part of this document may be photocopied, reproduced, or translated into another language without a legal license agreement from IBM Corporation.

Objectives

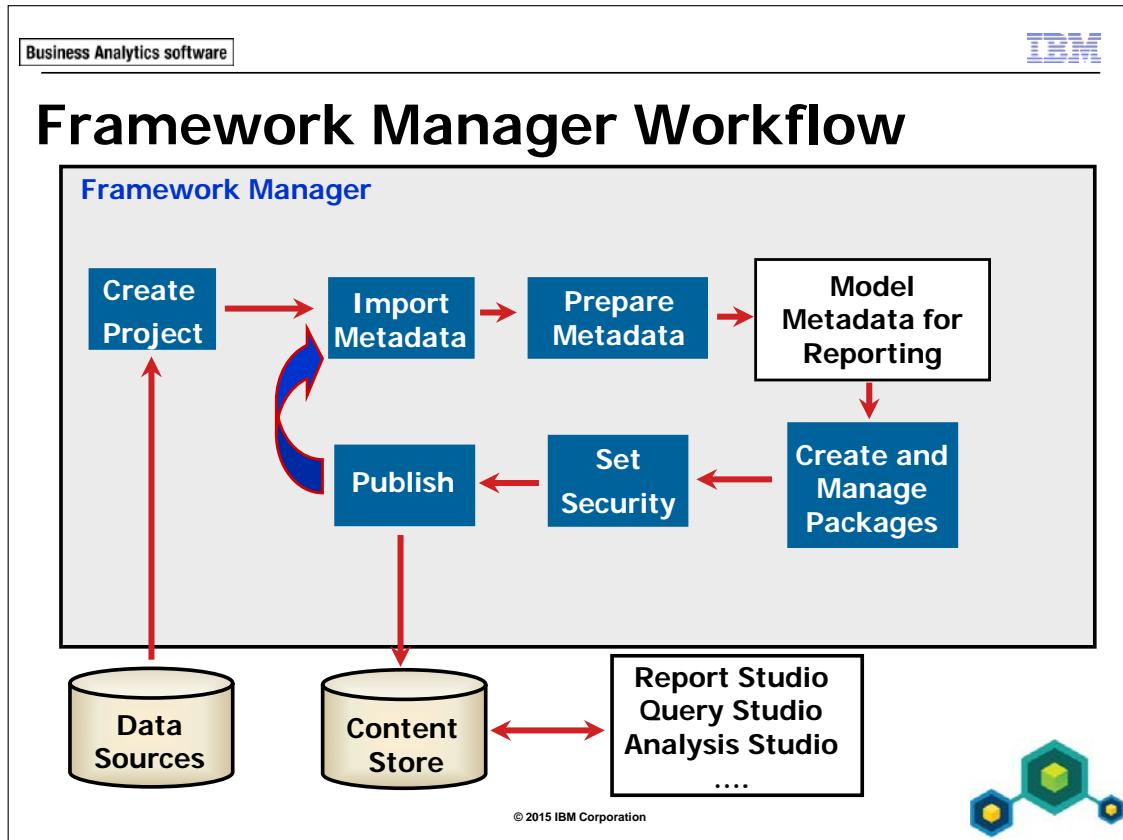
- At the end of this module, you should be able to:
 - use calculations to create commonly-needed query items for authors
 - use static filters to reduce the data returned
 - use macros and parameters in calculations and filters to dynamically control the data returned

A faint background image of a document page with a red circular stamp and a green ribbon or seal across it.

© 2015 IBM Corporation

Unless otherwise specified in demo or workshop steps, you will always log on to IBM Cognos in the Local LDAP namespace using the following credentials:

- User ID: admin
- Password: Education1



This module deals with creating calculations and filters to add business logic to the model.

Creating Calculations

- create calculations to provide report authors with values that they regularly use
 - Revenue = Quantity * Unit Sale Price
- calculations can use query items, parameters, and functions
- two types of calculations:
 - embedded
 - stand-alone

© 2015 IBM Corporation



If you want to create a calculation specifically for one query subject or dimension, you can embed the calculation directly in that object. For query subjects, this calculation can be done for either data source query subjects or model query subjects. However, it is recommended that you apply calculations in model query subjects wherever possible. This allows for better maintenance and change management.

Create a stand-alone calculation when you want to apply the calculation to more than one query subject or dimension. Stand-alone calculations are also valuable if you need to do aggregation before performing the calculation (as shown in Appendix A). You can also create stand-alone calculations as an alternate way to present information rather than in a query subject or dimension. However, stand-alone calculations are not visible in Analysis Studio.

If you start with an embedded calculation, you can later convert it into a stand-alone calculation that you can apply to other query subjects.

Demo 1: Create Embedded Calculations

Purpose:

Report authors want to include the revenue, gross profit, and margin of each order in their reports. Therefore, you will create three embedded calculations in the Sales Fact query subject. This will produce three new query items.

Component: **Framework Manager**

Project: **GO Operational**

Task 1. Create the calculations.

1. In **Framework Manager**, open the **GO Operational** model located at **C:\Edcognos\B5A52\CBIFM-Start Files\Module 9\GO Operational**.
2. Expand **Foundation Objects View > gosales**, and then double-click **Sales Fact**.

The calculations you are about to create could also be placed in the Consolidation View Sales Fact model query subject. However, placing them in the lower level increases opportunities for reusability.

3. Click **Add** in the bottom right corner.
4. In the **Name** box, type **Revenue**, and then, in the **Available Components** pane, double-click **Quantity**.
5. Click the **Functions** tab, expand **Operators**, and then double-click the multiplication operator (*).

Notice there is a description of the function in the Tips pane.

6. Under **Available Components**, click the **Model** tab, and then double-click **Unit Sale Price**.

The results appear as follows:

[gosales].[Sales Fact].[Quantity] * [gosales].[Sales Fact].[Unit Sale Price]

7. Click **Test Sample**  to verify that the calculation works, and then click **OK**.

8. Click **Add**, and create a **Gross Profit** calculation with the following expression:

$$[\text{gosales}].[\text{Sales Fact}].[\text{Revenue}] - ([\text{gosales}].[\text{Sales Fact}].[\text{Quantity}] * [\text{gosales}].[\text{Sales Fact}].[\text{Unit Cost}])$$

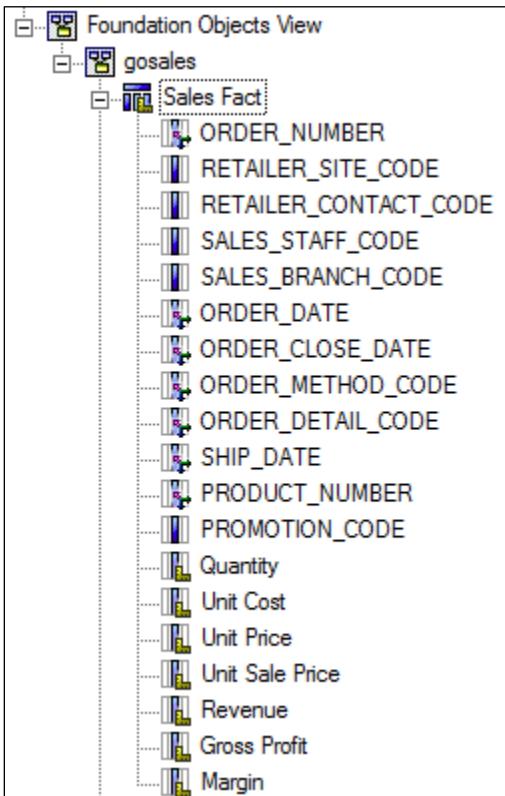
Note: Revenue is the calculation you just created.
9. Click **Add**, and create a **Margin** calculation with the following expression:

$$[\text{gosales}].[\text{Sales Fact}].[\text{Gross Profit}] / [\text{gosales}].[\text{Sales Fact}].[\text{Revenue}]$$

Where Gross Profit and Revenue are the calculations you just created.

Currently the new calculations appear as attributes because they have not been evaluated yet.
10. In the **Query Subject Definition** dialog, click **Validate**, and then click **OK**.

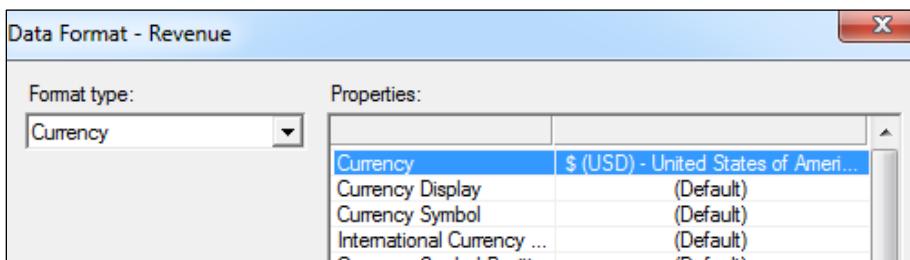
The query subject and its new calculations are evaluated.
The results appear as follows:



The new calculations (Revenue, Gross Profit, and Margin) now appear as facts.

Task 2. Set formatting properties on query items in the Foundation Objects View.

1. In **Foundation Objects View > gosales > Sales Fact**, select **Revenue** and **Gross Profit**.
If necessary, from the **View** menu, click **Properties**.
2. In the **Properties** pane, for the **Revenue** query item, click the cell under **Format**.
3. Set **Format type** to **Currency**, and the **Currency** property to **\$ (USD) - United States of America, dollar**.



4. Click **OK**, and then apply the same formatting to **Gross Profit** by dragging down the small black arrow.
5. In the **Project Viewer**, click **Margin**, and then change the **Format type** to **Percentage**.
6. Click **OK**.

Task 3. Update the Consolidation View Sales Fact query subject and test.

1. In **Consolidation View**, double-click **Sales Fact**.
2. Navigate to **Foundation Objects View > gosales > Sales Fact**, and drag **Revenue**, **Gross Profit**, and **Margin** to the right pane.
3. Click **Revenue**, and then click **Top** to move it to the top of the list.

4. Click the **Test** tab, and then click **Test Sample**.

The results appear as follows:

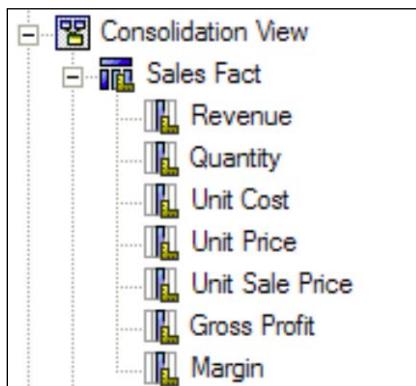
Revenue	Quantity	Unit Cost	Unit Price	Unit Sale Price	Gross Profit	Margin
8624.64	256	15.62	35.09	33.69	4625.92	0.53636093796379
9411.6	92	49.69	110	102.3	4840.12	0.51427174975562
18032.22	162	80	119.69	111.31	5072.22	0.28128649717007
6690.8	172	23.53	40.52	38.9	2643.64	0.39511568123393
24747.82	74	176.47	359.6	334.43	11689.04	0.47232604730437
6825.6	90	39	81.55	75.84	3315.6	0.48575949367089
2532	422	2.76	6	6	1367.28	0.54
21170.52	3252	2.42	7	6.51	13300.68	0.62826420890937

Notice that the Margin values are equal to Gross Profit divided by Revenue, as expected.

Note: If you use auto sum you will not get the expected results for the Margin calculation. This is because the calculation is performed first and then aggregated (in this case summed). For this type of calculation, you need to aggregate first and then perform the calculation. This can only be done with a stand-alone calculation. This technique is covered towards the end of the course in the Additional Modeling Techniques module.

5. Click **OK**.

The results appear as follows in the Project Viewer:



6. Save the project.

Results:

You created and tested three embedded calculations within the Sales Fact query subject. This produced three new query items that report authors can use.

Embedded filters are appropriate when the filter is intended for just one query subject or dimension. Stand-alone filters are appropriate when required in multiple query subjects or dimensions, or to make commonly used filters readily available for authors.

Filters have a Usage setting with the following options:

- Always - the filter will always be applied, regardless of whether the filtered query item is in the query or not
- Optional - users may choose to enter a filter value or leave it blank (only applies to filters that use a prompt value or macro)
- Design Mode Only - Limits the amount of data that is retrieved when testing in Framework Manager or when authoring reports.

You can also restrict the data that a query retrieves by adding a WHERE clause to the SQL definition, or by setting governors. Governors are discussed in the Optimize and Tune Framework Manager Models module.

Demo 2: Create Embedded and Standalone Filters

Purpose:

Employee data includes historical job information that is not required by report authors. Therefore you will restrict EMPLOYEE_HISTORY to each employee's current record.

As well, report authors want to be able to restrict retailer data to specific regions. To do this, you will create standalone filters that authors can apply during report creation.

Components: Framework Manager, IBM Cognos Workspace Advanced
 Project: GO Operational
 Package: GO Operational

Task 1. Create an embedded filter to retrieve current employee records.

To retrieve current employee records, you will filter data by EMPLOYEE_HISTORY.RECORD_END_DATE IS NULL. You will place the filter on EMPLOYEE_HISTORY in the Foundation Objects View (rather than on Staff by Location in the Consolidation View) because the filter causes the relationship between EMPLOYEE and EMPLOYEE_HISTORY to change from one-to-many to one-to-one, and all our relationships are in the Foundation Objects View. Changing this cardinality also prevents EMPLOYEE_HISTORY from acting as an ambiguous query subject.

1. In the **Project Viewer** pane, expand **Foundation Objects View > gosales**, and then double-click **EMPLOYEE_HISTORY**.
 EMPLOYEE_HISTORY can act as a fact in certain query scenarios because of its cardinalities that can cause unwanted query splits.
2. Click the **Filters** tab, click **Add**, and then in the **Name** box, type **Current Employee History Record**.
3. In the **Available Components** pane, double-click **RECORD_END_DATE**.

4. Click in the **Expression Definition** pane at the end of the expression, and then type **IS NULL**.

Note: Be sure to type IS NULL and not =Null, the difference is subtle, but important.

The results appear as follows:

[gosales].[EMPLOYEE_HISTORY].[RECORD_END_DATE] IS NULL

5. Click **OK**, click the **Test** tab, and then click **Test Sample**.

The results are similar to the following:

Test results				
Y_PARENT	EMPLOYEE_CODE	RECORD_START_DATE	RECORD_END_DATE	POSITION
	10004	Dec 11, 2007 12:00:00 AM		4500
	10005	Nov 24, 2009 12:00:00 AM		5700
	10006	May 10, 2012 12:00:00 AM		5500
	10007	Oct 9, 2009 12:00:00 AM		5600
	10012	Mar 22, 2011 12:00:00 AM		5400
	10013	Nov 7, 2006 12:00:00 AM		5300
	10014	May 12, 2009 12:00:00 AM		5700
	10015	Aug 10, 2012 12:00:00 AM		5600

Only one record for every employee is returned and each has a null RECORD_END_DATE.

6. Click **OK**.
7. Right-click **EMPLOYEE_HISTORY** and click **Launch Context Explorer**.
8. Click **Show Related Objects**, and then double-click the relationship between **EMPLOYEE_HISTORY** and **EMPLOYEE**.
9. Change the **EMPLOYEE_HISTORY** side to **1..1**.
This reflects the impact of the new filter.
10. Click **OK**, close the **Context Explorer**, and then **Save** the project.

Task 2. Create standalone Retailer Location filters.

1. In **Foundation Objects View > gosales**, right-click **SALES_REGION**, click **Test** and then click **Test Sample**.

The five regions are:

710 = Americas
 740 = Asia Pacific
 750 = Northern Europe
 760 = Central Europe
 770 = Southern Europe

2. Click **Close**.
3. In the **Consolidation View**, create a new folder called **Model Filters** without selecting data.

You will create filters in the Consolidation View rather than the Foundation Objects View because the filters you are creating are going to be used in the final presentation view

4. In the new **Model Filters** folder, create a new folder called **Retailer Location Filters** without selecting data.
5. Right-click **Retailer Location Filters**, point to **Create**, and then click **Filter**.
6. In the **Name** box, type **Americas**.
7. In the **Available Components** pane, under **Consolidation View**, expand **Retailer by Location > Codes**.
8. Add **Retailer Site Region Code** to the **Expression definition** pane, and then type **= 710**.

The results appear as follows:

[Consolidation View].[Retailer by Location].[Retailer Site Region Code] = 710

9. Click **OK**.

10. Repeat steps **5** to **8** to create new filters using the appropriate **Retailer Site Region Code**.

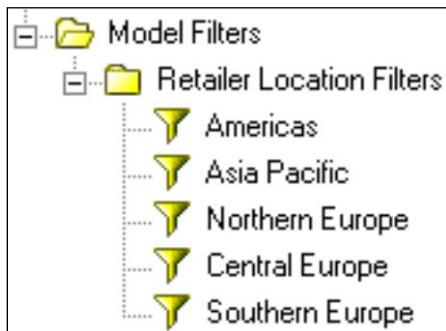
Retailer Site	Retailer Site Region Code
Asia Pacific	740
Northern Europe	750
Central Europe	760
Southern Europe	770

Tip: You can also copy the Americas filter and simply edit it to meet your needs.

The expression for Asia Pacific will appear as follows:

[Consolidation View].[Retailer by Location].Retailer Site Region Code] = 740

The results appear as follows:



11. **Save** the project.

Task 3. Test the standalone filters.

1. Publish the **GO Operational** package.
2. In **IBM Cognos Connection**, open **IBM Cognos Workspace Advanced**, and then select the **GO Operational** package to create a **List** report.
3. In the **Source** pane, expand **Consolidation View > Retailer by Location**, and then drag **Retailer Site Country** onto the report.

4. Expand **Model Filters > Retailer Location Filters**, and then drag **Americas** onto the report.

The results appear as follows:



The report is limited to records for the American continents.

5. Close your browser without saving the report, and leave Framework Manager open for the next demo.

Results:

You embedded a filter into EMPLOYEE_HISTORY to restrict the data to current employees. You also created stand-alone filters for retailer locations that authors can apply during report creation.

Customizing Metadata for Run Time

- modify query subjects to dynamically control the data returned using:
 - session parameters
 - parameter maps
 - macros

© 2015 IBM Corporation



This concept can be used to dynamically return data from specific columns or rows, and even from specific data sources.

One example is to dynamically implement security by using a calculation to retrieve a user's account, group, or role information and then implement row-level security based on values stored in the data source.

You could also retrieve location specific data for a particular user. For example, if a user works in Mexico, you can use a calculation to return sales values for Mexico by querying the Revenue_Mexico column rather than other columns in the table such as Revenue_Canada or Revenue_Germany.

IBM Cognos Environment Session Parameters

- predefined and stored in the content store database

Parameter	Value	Override Value
account.defaultName	Bart Scott	
account.parameters.Location	Seattle	
account.personalInfo.businessPhone	1 (206) 292-0012	
account.personalInfo.email	BScott@grid123.com	
account.personalInfo.faxPhone	1 (206) 292-3312	
account.personalInfo.givenName	Bart	
account.personalInfo.surname	Scott	
account.personalInfo.userName	scottb	
runLocale	en	

© 2015 IBM Corporation



A session parameter returns session information at run time (for example, runLocale or account.UserName).

Business Analytics software

IBM

Parameter Maps

- used to substitute one value for another
- keys must be unique

Key	Value
Bart Scott	60
Ana Orozco	80
Alex Rodriguez	53

Source Values **Target Values**



© 2015 IBM Corporation

A parameter map is a two-column table that maps a set of keys (source) to a set of substitution values. The slide example uses a parameter map to convert the account.DefaultName session parameter to a value in the data (staff key).

In DQM-enabled projects, Stored Procedures have a Freshness property that controls how often parameter values are refreshed. This lets you balance the performance benefits of caching with your data latency requirements. This property can be set to one of the following values:

Setting	Description
-1	Parameters are not reloaded during the session.
0	Parameter values are reloaded whenever a query is initiated that references the parameter map.
>0	Parameter values are reloaded after the specified number of seconds.
Macro	Refresh frequency is dynamically determined by resolving a macro value. For example: #case \$Name when 'A' then 2 else -1 end#.

Macros

Filter Expression

```
[Sales Target Fact].[Sales Staff Code] =  
#$$SecurityLookup{$account.defaultName}#
```

Parameter Map

Key	Value
Bart Scott	60
Ana Orozco	80
Alex Rodriguez	53

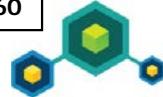
Session Parameter
passed to parameter map

Macro

Generated SQL

```
.....  
Where  
Sales_Target_Fact.Sales_Staff_Code = 60
```

© 2015 IBM Corporation



A macro is a fragment of code that you can insert within filters, calculations, properties, and so on, that are to be evaluated at run time. Macros are enclosed by the # character.

You will use the technique illustrated in the slide later in the course when applying security.

Business Analytics software

IBM

Dynamically Retrieve Language Column

Region calculation =
'SALES_REGION_' +
a string based on
user's locale setting

1. Call runLocale session parameter: en, en-uk, fr, ...
2. Use a parameter map to convert to EN, FR, ...
3. Concatenate the results with a string to form the desired column name
 - SALES_REGION_EN
 - SALES_REGION_FR ...

© 2015 IBM Corporation

In the next demo, you will dynamically return a specific column in the data source based on a user's locale.

Instead of placing a macro in a calculation or filter, you can also edit the SQL directly in a query item; however this may affect the query engine's ability to properly minimize the SQL. Modelers and Report Authors should be aware of this if they are examining the generated SQL in Framework Manager or Report Studio.

Demo 3: Calculate Based on Local Language

Purpose:

Report authors want report consumers to automatically view region names in the language of their locale. To enable this, you will replace Retailer Site Region with a calculation that references a parameter map and a session parameter. You will make this calculation standalone, so that you can reuse it for Branch Region, Staff Region, and Sales Target by Location.

Component: **Framework Manager**

Project: **GO Operational**

Task 1. Create a parameter map.

1. In the **Project Viewer** pane, **Foundation Objects View > gosales**, expand **SALES_REGION**.

This query subject has column-based multilingual data. This allows you to select the language in which you want to view the data.

2. In **Foundation Objects View > gosales**, double-click **Retailer Site Region (alias)**.

This model query subject was created with just one language query item, **SALES_REGION_EN**. You want to change this to pick the underlying query item based on a user's locale session parameter at run time. For example, the session locales for United States and France are **en-us** and **fr-fr**.

3. Click **Cancel**.

You will now create a parameter map to substitute a locale session parameter to a value found in the data.

4. Right-click **Parameter Maps**, point to **Create**, and then click **Parameter Map**.

You can manually type in the values or import them from a file. You can also base the parameter map on existing query items (i.e. a table that maps locales to language values in other tables). If so, consider filtering the query subject based on the session parameter you are using to look up the value (i.e. **runLocale**).

This reduces the record set to one record, rather than forcing a scan of the entire table at run time. For example, if the value is en-us at run time, the query subject will only return one row consisting of en-us and EN (the key field and the value field) instead of all rows. In other words, you are dynamically reducing the parameter map list to one record before accessing it for a value.

For this exercise there is no table that contains these mapping values, so you will use a text file that has the mappings entered for you.

5. In the **Name** box, type **Language_lookup**, and then click **Next**.
6. Click **Import File**.
7. Navigate to **C:\Program Files (x86)\IBM\cognos\c10\webcontent\samples\models**, click **Language_lookup.txt**, and then click **Open**.
8. Click **OK**.
- The values in the file are imported. Note that 'en-us' (and all other English variants) map to 'EN'. The same applies for other languages and their locales.
9. Click **Finish**.

Task 2. Create an embedded calculation for multilingual data.

1. In **Foundation Objects View > gosales > Retailer Site Region (alias)**, double-click **Retailer Site Region**.
 2. In the **Expression definition** pane, to the left of **[gosales].[SALES_REGION].[SALES_REGION_EN]**, type '#'
 3. Replace **EN**] with ' +
- The results appear as follows:
- ```
#'[gosales].[SALES_REGION].[SALES_REGION_] +
```
4. Under **Available Components**, click the **Parameters** tab, expand **Parameter Maps**, and then double-click **Language\_lookup**.
  5. Collapse **Language\_lookup**, expand **Session Parameters**, and then double-click **runLocale**.

The **runLocale** session parameter will convert the string to the language code for the current session user's locale setting (i.e. EN), which will be merged with the SALES\_REGION\_ prefix and the ] suffix to create the desired query item name.

- At the end of the expression, type + "]'#"

The results appear as follows:

```
#'[gosales].[SALES_REGION].[SALES_REGION_'] +
$Language_lookup{$runLocale} + ']'#
```

The # character at the beginning and the end of the expression identify this expression as a macro.

Note: This technique hard codes the path to your source query item. If you move or rename the source, you will have to update this calculation manually.

- Click **Test Sample**  to verify that the calculation works for your current locale, and then click **OK**.

The Sales Region data is displayed in the language set in the user's locale.

- Save** the project.

### Task 3. Test the calculation under a different language.

- From the **Project** menu, point to **Languages**, and then click **Define Languages**.
- Select **French**, click the arrow to add it to the list of **Project languages**, and then click **OK** twice.
- If the **Tools** pane is not displayed, from the **View** menu, click **Tools**.
- On the **Summary** tab, change **Active Language** to **French**.  
Notice that all object names in the Project Viewer are preceded by "(fr)". This is to remind you to translate the strings if required.
- In **(fr) Foundation Objects View > (fr) gosales**, right-click **(fr) Retailer Site Region (alias)**, click **Test**, and then click **Test Sample**.

The results appear as follows:

| Test results                   |                           |
|--------------------------------|---------------------------|
| (fr) Retailer Site Region Code | (fr) Retailer Site Region |
| 710                            | Amériques                 |
| 740                            | Asie-Pacifique            |
| 750                            | Europe septentrionale     |
| 760                            | Europe centrale           |
| 770                            | Europe méridionale        |

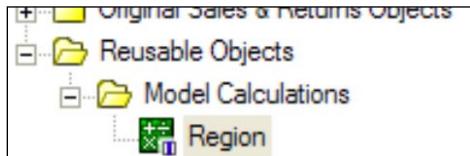
The names appear in French based on the macro you created.

- Click **Close**, and then, in the **Tools** pane, change the **Active Language** back to **English**.

## Task 4. Make the calculation reusable.

1. In **Foundation Objects View > gosales**, double-click **Retailer Site Region (alias)**.
2. In **Query Items and Calculations**, right-click **Retailer Site Region**, and then click **Convert to Stand-alone Calculation**.
3. Click **OK**, and then, at the bottom of the **gosales** namespace, double-click the new **Retailer Site Region** calculation.  
The calculation has been preserved.
4. Click **Cancel**.
5. In **Foundation Objects View > gosales**, create a folder called **Reusable Objects** without selecting data.
6. In the **Reusable Objects** folder, create a folder called **Model Calculations** without selecting data.
7. Drag the **Retailer Site Region** calculation into the **Model Calculations** folder.
8. Rename the calculation to **Region**.

The results appear as follows:

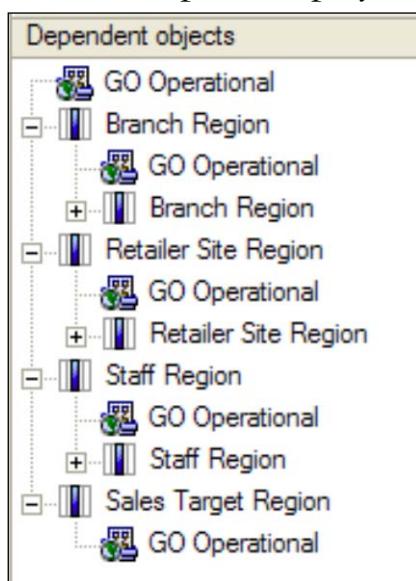


## Task 5. Apply the calculation to Branch Region (alias).

1. In the **Foundation Objects View > gosales**, expand **Branch Region (alias)**, and then double-click **Branch Region**.  
The expression uses the English region column. Rather than recreate the more generic language-based version, you will reuse the Region calculation.
2. Delete the expression in the **Expression definition** box.
3. In the **Available Components** pane, expand **Reusable Objects > Model Calculations**, and then double-click **Region**.
4. Test the expression, and then click **OK**.

5. Repeat steps **1** to **3** to apply the **Region** calculation to the following query items:
  - **Foundation Objects View > Staff Region (alias) > Staff Region**
  - **Consolidation View > Sales Target by Location > Sales Target Region**
 To see if you applied the calculation in all the required locations, you can use the Show Object Dependencies feature.
6. In **Foundation Objects View > gosales > Reusable Objects > Model Calculations**, right-click the **Region** calculation, and then click **Show Object Dependencies**.

The Tools pane displays a list of dependent objects.



You can click any of these objects to give them focus in the Project Viewer or Diagram pane. By doing this, you can verify that you applied the calculation in all required areas.

Note: You could have placed the calculation directly in the SALES\_REGION data source query subject, but calculations in data source query subjects will cause additional queries to the data source for metadata. For some data source vendors this can have a performance impact. This also affects portability of the model and change management.

## Results:

**You modified Retailer Site Region (alias) to automatically display region names in the language identified in the report consumer's locale setting. You then made the calculation stand-alone, and reused it for Branch Region, Staff Region, and Sales Target Region.**

Business Analytics software

IBM

## Filtering by Locale

| Product Number | Product Language | Product Name  |
|----------------|------------------|---------------|
| 99110          | DE               | Aloe Balsam   |
| 99110          | DA               | Aloe Lindring |
| 99110          | NO               | Aloe lotion   |
| 99110          | EN               | Aloe Relief   |

**PRODUCT\_LANGUAGE Filter =  
a string based on user's locale setting**

1. Call runLocale session parameter: en, en-uk, de, ...
2. Use a parameter map to convert to EN, DE, ...
3. Use sq macro function to wrap in single quotes: 'EN', 'DE', ...

© 2015 IBM Corporation



In addition to using a calculation to select column-based language values, you can use a filter to selects row-based language values. However, instead of using a string to build the column name, you will create a string that matches a PRODUCT\_LANGUAGE row value and filter on this value.

## Workshop 1: Filter Locale Language

Each product in the PRODUCT\_NAME\_LOOKUP table has multiple records (one name per language). To ensure that report consumers will view the product names in the language of their locale, you will add a filter to PRODUCT\_NAME\_LOOKUP that equates PRODUCT\_LANGUAGE to #sq( \$Language\_lookup {\$runLocale} ) #

To do this:

- Use the **runLocale** session parameter to extract the locale.
- Convert the locale to the format of our **PRODUCT\_LANGUAGE** column by using the **Language\_lookup** parameter map that you created earlier.
- Wrap the string in single quotes by using the **sq** macro.

Note that the filter changes the nature of the relationship between PRODUCT\_NAME\_LOOKUP and Product Type & Product to be 1..1 on both sides. Applying this filter and changing the cardinality will resolve unexpected results when querying Product Name.

- In the **Consolidation View**, test **Product > Product Name**, and **Sales Fact > Revenue**.
- Save the project when finished.

**Tip:** You will need to use the sq macro function, because once the macro resolves to the appropriate value from the parameter map it must be wrapped in quotes in order to be identified as a string by the filter.

For more information about where to work and the workshop results, refer to the Tasks and Results section that follows. If you need more information to complete a task, refer to earlier demos for detailed steps.

## Workshop 1: Tasks and Results

### Task 1. Add a filter to PRODUCT\_NAME\_LOOKUP.

- In Foundation Objects View > gosales, add a Filter to **PRODUCT\_NAME\_LOOKUP**, named **Language Filter**.
- From Foundation Object View > gosales > **PRODUCT\_NAME\_LOOKUP**, drag **PRODUCT\_LANGUAGE** to the expression.
- Type **=** at the end of the expression.
- Select the **Parameters** tab, expand **Macro Functions**, and then double-click **sq**.
- Expand **Parameter Maps** and double-click **Language\_lookup**.
- Expand **Session Parameters** and double-click **runLocale**.

The results appear as follows:

**[gosales].[PRODUCT\_NAME\_LOOKUP].[PRODUCT\_LANGUAGE]**  
**= #sq(\$Language\_lookup{\$runLocale})#**

- Click **OK** then test the filter using the **Test** tab.

The results appear as follows:

| Test results   |                  |                                |                                                                                                     |
|----------------|------------------|--------------------------------|-----------------------------------------------------------------------------------------------------|
| PRODUCT_NUMBER | PRODUCT_LANGUAGE | PRODUCT_NAME                   |                                                                                                     |
| 1110           | EN               | TrailChef Water Bag            | Lightweight, collapsible bag to carry liquids easily. Wide mouth for easy filling. Holds 10 liters. |
| 2110           | EN               | TrailChef Canteen              | Aluminum canteen. Rugged fleece-lined cover with belt clips, removable shoulder sling and sr        |
| 3110           | EN               | TrailChef Kitchen Kit          | Zippered nylon pouch contains cutlery for two, can opener, wire whisk, scrubber sponge, 2 6         |
| 4110           | EN               | TrailChef Cup                  | Tin cup. Holds 0.4 liters. Weight: 60 g                                                             |
| 5110           | EN               | TrailChef Cook Set             | All you will ever need on the trail. Pot gripper and nylon carrying bag included. 1.5 and 2 liter   |
| 6110           | EN               | TrailChef Deluxe Cook Set      | Cascade set features 1, 2, and 3 liter pots with individual lids that double as fry pans or plates. |
| 7110           | EN               | TrailChef Single Flame         | A single burner stove with integral tank. It features precise flame control so you will never hav   |
| 8110           | EN               | TrailChef Double Flame         | The TrailChef Double Flame is a camp stove which attaches directly to fuel canisters. Feature       |
| 9110           | EN               | TrailChef Kettle               | This old-fashioned style kettle is made of durable enamel. Weight: 350g, Holds 1.5 liters.          |
| 10110          | EN               | TrailChef Utensils             | Spoon, fork and knife set made of a light yet strong metal alloy. Carrying pouch included.          |
| 11110          | EN               | Star Lite                      | A perfect tent for biking and hiking trips, compact and very light. Features three shock cord a     |
| 12110          | EN               | Star Dome                      | Four pole geodesic dome mountain tent that comfortably fits two to three people. 10 mm alumi        |
| 13110          | EN               | Star Gazer 2                   | The Star Gazer 2 is the Great Outdoor's most versatile tent. It is a large two-person tent that is  |
| 14110          | EN               | Star Gazer 3                   | The Star Gazer 3 features a water proof fly, mesh window and two doors for good ventilation.        |
| 15110          | EN               | Star Gazer 6                   | Massive cabin tent has ample room for six people. Mesh door and vent, T-style door. Four sid        |
| 16110          | EN               | Star Peg                       | A single tent peg made of heavy-duty plated steel. Long enough even for hard, rocky ground.         |
| 17110          | EN               | Hibemator Lite                 | The Hibemator Lite is the perfect summer sleeping bag. Light weight and comfortable down ti         |
| 18110          | EN               | Hibemator                      | The Hibemator is a three-season sleeping bag. The rectangular shape allows for easy pairing.        |
| 19110          | EN               | Hibemator Extreme              | The Hibemator Extreme has all the features you need for four-season comfort. Comfortable fr         |
| 20110          | EN               | Hibemator Self - Inflating Mat | Mattress is made of extra-thick foam. Mesh stuff sack included for easy carrying and storage.       |
| 21110          | EN               | Hibemator Pad                  | Warms up quickly for snow camping comfort. The Hibemator Pad will resist cracking in tempe          |
| 22110          | EN               | Hibemator Pillow               | Camp pillow is filled with soft, luxurious fibers for maximum compactness and resiliency. Covere    |
| 23110          | EN               | Hibemator Camp Cot             | Aluminum frame camp cot that is lightweight and durable. Size: 100 x 225 x 50 cm. Weight: :         |
| 24110          | EN               | Canyon Mule Climber Backpack   | This pack is perfect for day trips and short hikes. Also great for students. Separate front comp    |
| 25110          | EN               | Canyon Mule Weekender Backpack | A weekend getaway requires this pack. It features a large front compression pocket, harness         |

This material is meant for IBM Academic Initiative use only. NOT FOR RESALE

## Task 2. Update the relationship.

- Change the relationship between **PRODUCT\_NAME\_LOOKUP** and **Product Type & Product** to be 1..1 on both sides.
- In the **Consolidation View**, select and **Test** the following:
  - **Products.Product Name**
  - **Sales Fact.Revenue**

The results appear as follows:

| Product Name        | Revenue  |
|---------------------|----------|
| Firefly Mapreader   | 10572.64 |
| Flicker Lantern     | 8624.64  |
| Polar Ice           | 9411.6   |
| Edge Extreme        | 18032.22 |
| Bear Edge           | 6690.8   |
| Glacier GPS Extreme | 24747.82 |
| Mountain Man Deluxe | 6825.6   |
| Insect Bite Relief  | 2532     |
| BugShield Extreme   | 21170.52 |
| Sun Shield          | 6376.32  |

Product names are only returned in one language.

- **Save** the project.

## Workshop 2: Calculate Based on Local Language

In the last demo, you created a reusable Region calculation for Retailer Site Region, Branch Region, and Sales Target Region. You will now do the same thing for Retailer Site Country, Branch Country, and Sales Target by Location with a Country calculation.

Create a calculation as follows:

- Create a new calculation in **Foundation Objects View > gosales > Reusable Objects > Model Calculations**, called **Country**.
- Define the expression as:  
`#'[gosales].[COUNTRY].[COUNTRY_'] +  
$Language_lookup{$runLocale} + ']'#`
- Edit the following query items, so that they use the **Country** calculation:
  - **Foundation Objects View > Branch Country (alias) > Branch Country**
  - **Foundation Objects View > Retailer Site Country (alias) > Retailer Site Country**
  - **Foundation Objects View > Staff Country (alias) > Staff Country**
  - **Consolidation View > Sales Target by Location > Sales Target Country**
- **Save** and **Close** the project.

For more information about where to work and the workshop results, refer to the Tasks and Results section that follows. If you need more information to complete a task, refer to earlier demos for detailed steps.

## Workshop 2: Tasks and Results

### Task 1. Create a Country calculation.

- In **Foundation Objects View > gosales > Reusable Objects > Model Calculations**, create a **calculation** named **Country**, with the following expression:

```
#'[gosales].[COUNTRY].[COUNTRY_'
+$Language_lookup{$runLocale} + ']'#
```

### Task 2. Apply the calculation to additional query subjects.

- In the **Foundation View > gosales > Branch Country (alias)** query subject, edit **Branch Country**.
- Delete the existing expression definition.
- In **Available Components**, add the **Country** calculation to the **Expression definition**.
- Test the query subject, and then click **OK**.
- In the **Project Viewer**, right-click **Branch Country (alias)**, and select **Edit Definition**.

The results appear as follows:

| Query Items and Calculations: |                           |
|-------------------------------|---------------------------|
| Name                          | Source                    |
| Branch Country Code           | COUNTRY.COUNTRY_CODE      |
| Branch Region Code            | COUNTRY.SALES_REGION_CODE |
| <b>Branch Country</b>         | <b>Country</b>            |

The source for **Branch Country** has changed from the **COUNTRY.COUNTRY\_EN** query item, to the **Country** calculation.

- Repeat this process to add the **Country** calculation to the following query subjects:
  - Foundation View > **gosales** > **Retailer Site Country (alias)**
  - Foundation View > **gosales** > **Staff Country (alias)**
  - Consolidation View > **gosales** > **Sales Target by Location**

The results for the Retailer Site Country (alias) appear as follows:

| Query Items and Calculations: |                           |
|-------------------------------|---------------------------|
| Name                          | Source                    |
| Retailer Site Country Code    | COUNTRY.COUNTRY_CODE      |
| Retailer Site Region Code     | COUNTRY.SALES_REGION_CODE |
| <b>Retailer Site Country</b>  | <b>Country</b>            |

The results for the Staff Country (alias) appear as follows:

| Query Items and Calculations: |                           |
|-------------------------------|---------------------------|
| Name                          | Source                    |
| Staff Country Code            | COUNTRY.COUNTRY_CODE      |
| Staff Region Code             | COUNTRY.SALES_REGION_CODE |
| <b>Staff Country</b>          | <b>Country</b>            |

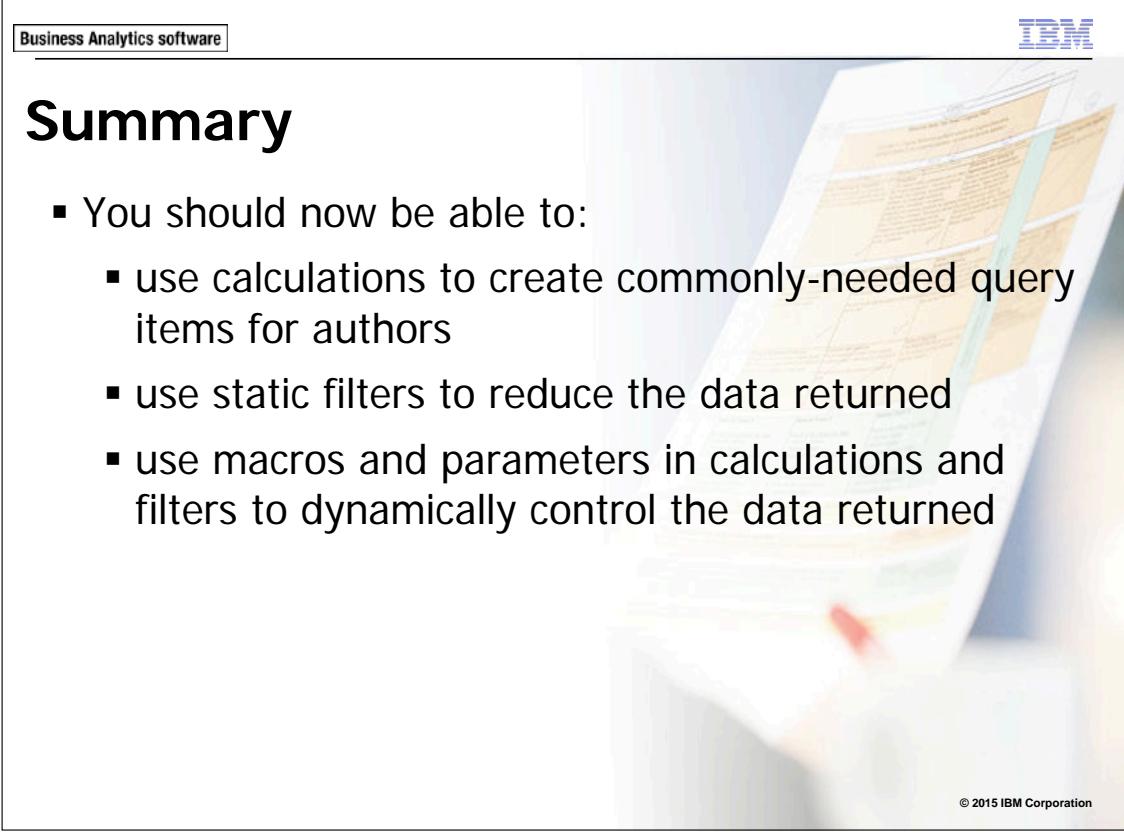
The results for Sales Target by Location appear as follows:

| Query Items and Calculations: |                                |
|-------------------------------|--------------------------------|
| Name                          | Source                         |
| Sales Target Region           | Region                         |
| <b>Sales Target Country</b>   | <b>Country</b>                 |
| Sales Target Region Code      | SALES_REGION.SALES_REGION_CODE |
| Sales Target Country Code     | COUNTRY.COUNTRY_CODE           |

- **Save** and **Close** the project.

## Summary

- You should now be able to:
  - use calculations to create commonly-needed query items for authors
  - use static filters to reduce the data returned
  - use macros and parameters in calculations and filters to dynamically control the data returned

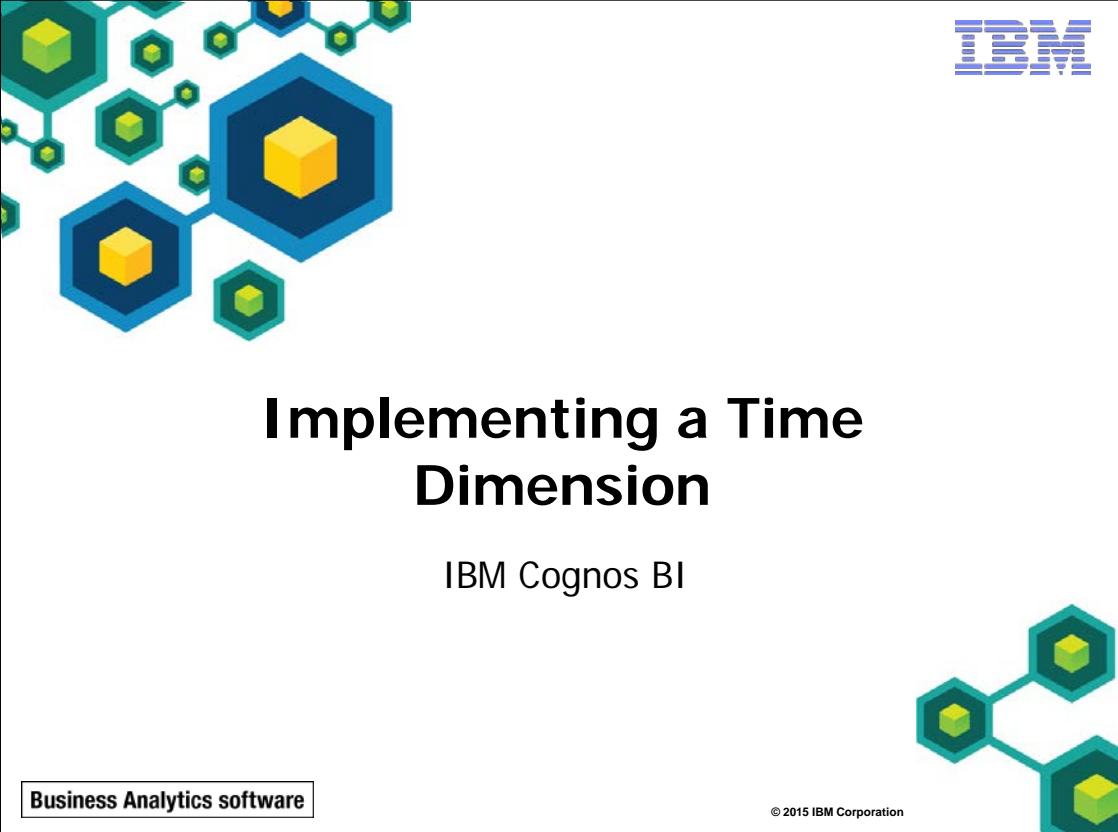


© 2015 IBM Corporation

This material is meant for IBM Academic Initiative use only. NOT FOR RESALE

© 2003, 2015, IBM Corporation

This guide contains proprietary information which is protected by copyright. No part of this document may be photocopied, reproduced, or translated into another language without a legal license agreement from IBM Corporation.



The slide features a white background with a decorative graphic of blue hexagons containing yellow cubes on the left side. In the top right corner is the IBM logo. Below the logo is a large, bold title. Underneath the title is the text "IBM Cognos BI". At the bottom left is a callout box with the text "Business Analytics software". At the bottom right is a small copyright notice.

**IBM**

# Implementing a Time Dimension

IBM Cognos BI

**Business Analytics software**

© 2015 IBM Corporation

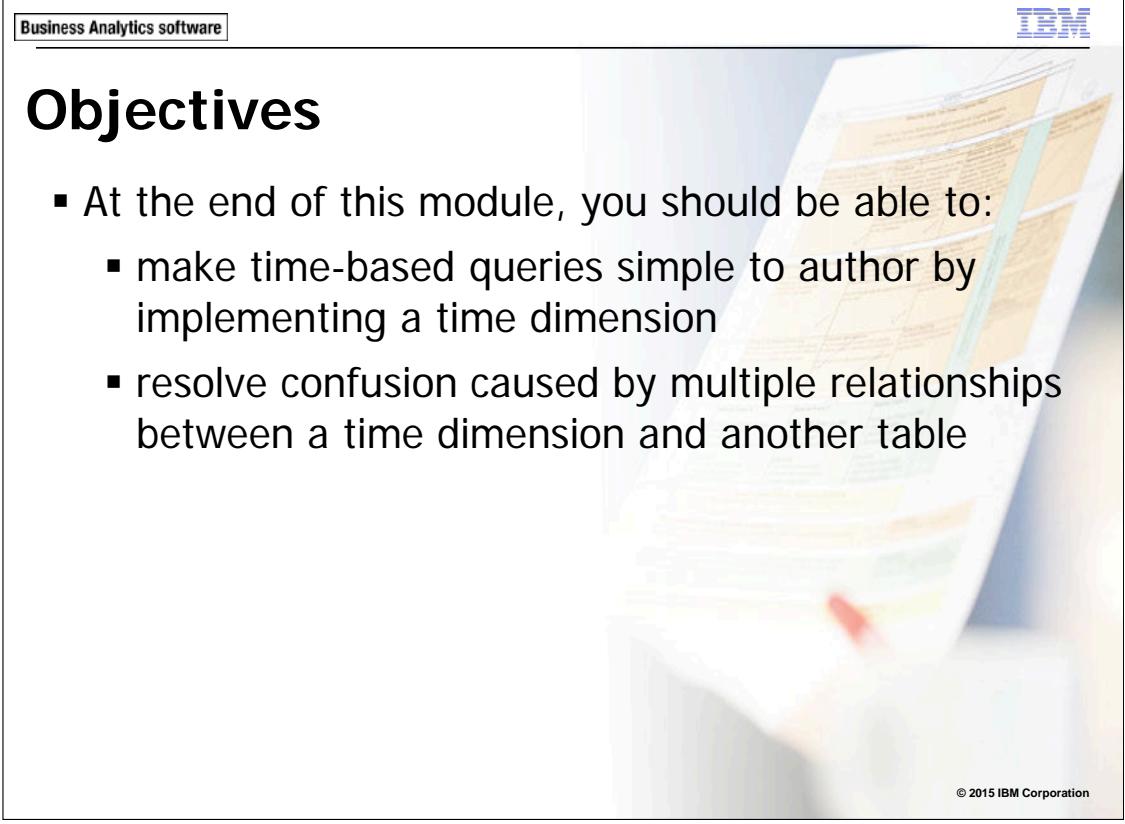
This material is meant for IBM Academic Initiative use only. NOT FOR RESALE

© 2003, 2015, IBM Corporation

This guide contains proprietary information which is protected by copyright. No part of this document may be photocopied, reproduced, or translated into another language without a legal license agreement from IBM Corporation.

# Objectives

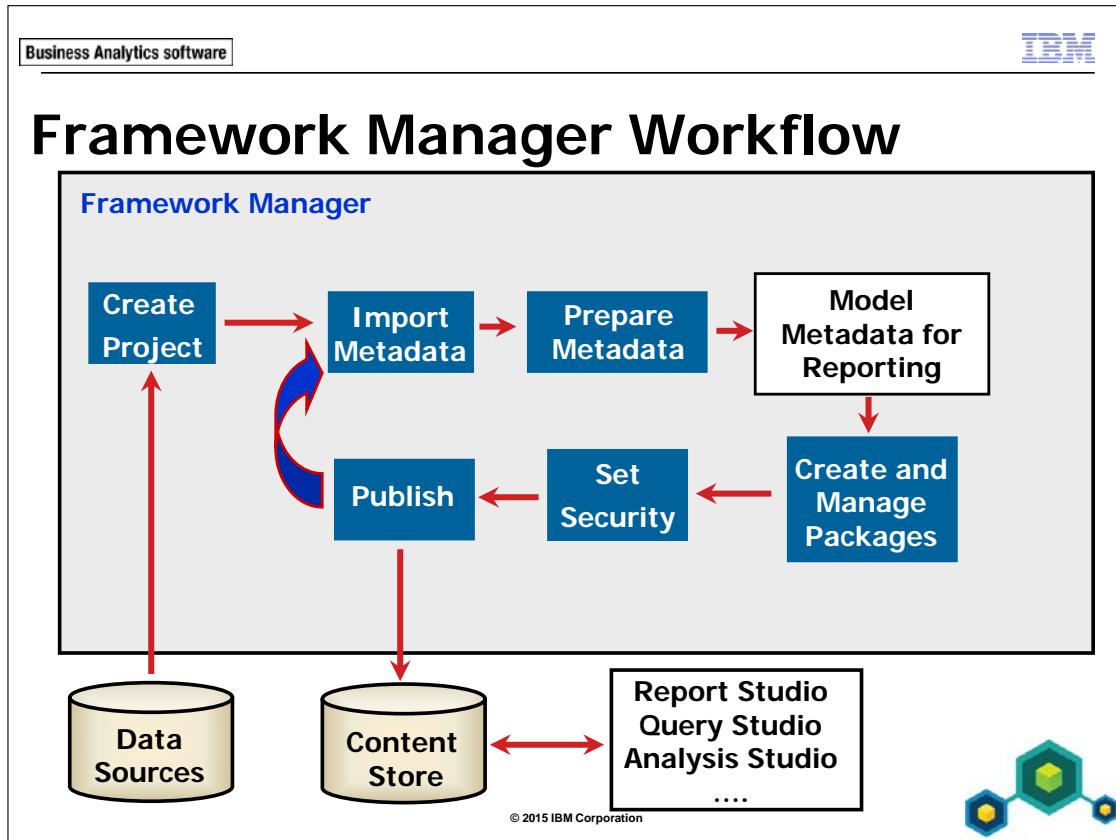
- At the end of this module, you should be able to:
  - make time-based queries simple to author by implementing a time dimension
  - resolve confusion caused by multiple relationships between a time dimension and another table



© 2015 IBM Corporation

Unless otherwise specified in demo or workshop steps, you will always log on to IBM Cognos in the Local LDAP namespace using the following credentials:

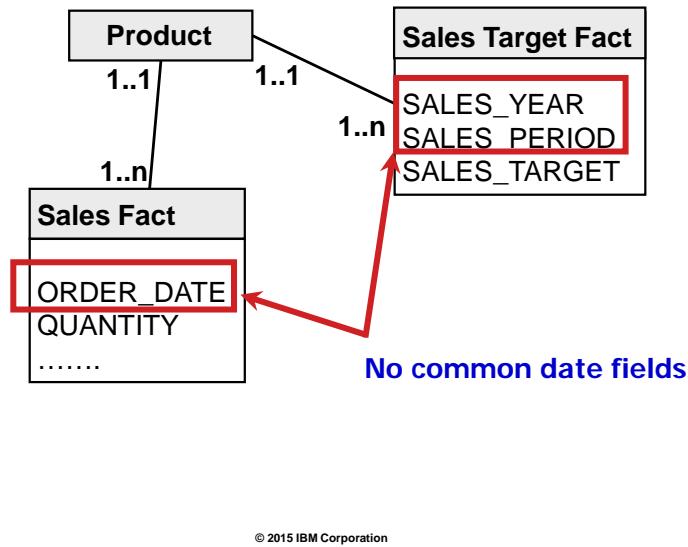
- User ID: admin
- Password: Education1



This module teaches the technique of implementing a time dimension in an operational model to simplify time-based queries across different facts.

## Report without a Time Dimension

- Reports require date calculations to match fields



When you do not have a Time dimension, you have to use date calculations to create matching values that let you compare data from different areas of the business. This can cause slower performance because it compares data on non-indexed values.

In the above example, you would need to create two calculations on ORDER\_DATE in the Sales Fact query subject in order to match the SALES\_YEAR and SALES\_PERIOD fields in the Sales Target Fact query subject. You will see this in the next demo.

As well, when date fields reside in fact tables the queries cannot be properly stitched together, since there is no conformed dimension.

## Demo 1: Report without a Time Dimension

### Purpose:

**Report Authors want to create a business report that compares the Monthly Sales Targets to Actual Sales. In this demo, you will examine whether this goal is easy to accomplish with the current metadata.**

Components: **Framework Manager, IBM Cognos Workspace Advanced**

Project: **GO Operational**

Package: **GO Operational**

Task 1. Examine problems when a time dimension is absent.

1. In **Framework Manager**, open the **GO Operational** project located at **C:\Edcognos\B5A52\CBIFM-Start Files\Module 10\GO Operational**. If prompted, log in as User ID **admin**, and Password **Education1**.
2. Publish the **GO Operational** package without verifying the package.
3. In **IBM Cognos Connection**, open **IBM Cognos Workspace Advanced**.
4. Select the **GO Operational** package, click **Create New**, and then double-click **List**.
5. Add the following items to the report from **Foundation Objects View**:

| Query Subject | Query Item                                                      |
|---------------|-----------------------------------------------------------------|
| Sales Fact    | <b>ORDER_DATE</b><br><b>Revenue</b>                             |
| SALES_TARGET  | <b>SALES_YEAR</b><br><b>SALES_PERIOD</b><br><b>SALES_TARGET</b> |

The results appear as follows:

| ORDER_DATE               | Revenue         | SALES_YEAR | SALES_PERIOD | SALES_TARGET |
|--------------------------|-----------------|------------|--------------|--------------|
| Jan 12, 2010 12:00:00 AM | \$39,036,796.40 | 2010       | 1            | 57,628,100   |
| Jan 13, 2010 12:00:00 AM | \$11,488,458.59 | 2010       | 2            | 67,795,500   |
| Jan 14, 2010 12:00:00 AM | \$3,232,160.48  | 2010       | 3            | 69,741,700   |
| Jan 15, 2010 12:00:00 AM | \$3,080,441.44  | 2010       | 4            | 59,954,700   |
| Jan 16, 2010 12:00:00 AM | \$1,976,896.69  | 2010       | 5            | 67,525,900   |
| Jan 19, 2010 12:00:00 AM | \$2,159,831.74  | 2010       | 6            | 73,594,900   |

The year and period (month) from SALES\_TARGET do not match those in the ORDER\_DATE from Sales Fact. You cannot get expected results without using a time dimension, creating calculations, or modeling complex summary queries that manipulate date fields to tie the data together.

For example, you could extract the year and month from ORDER\_DATE to match the fields in SALES\_TARGET using functions such as:

- Year(ORDER\_DATE)
- Month(ORDER\_DATE)

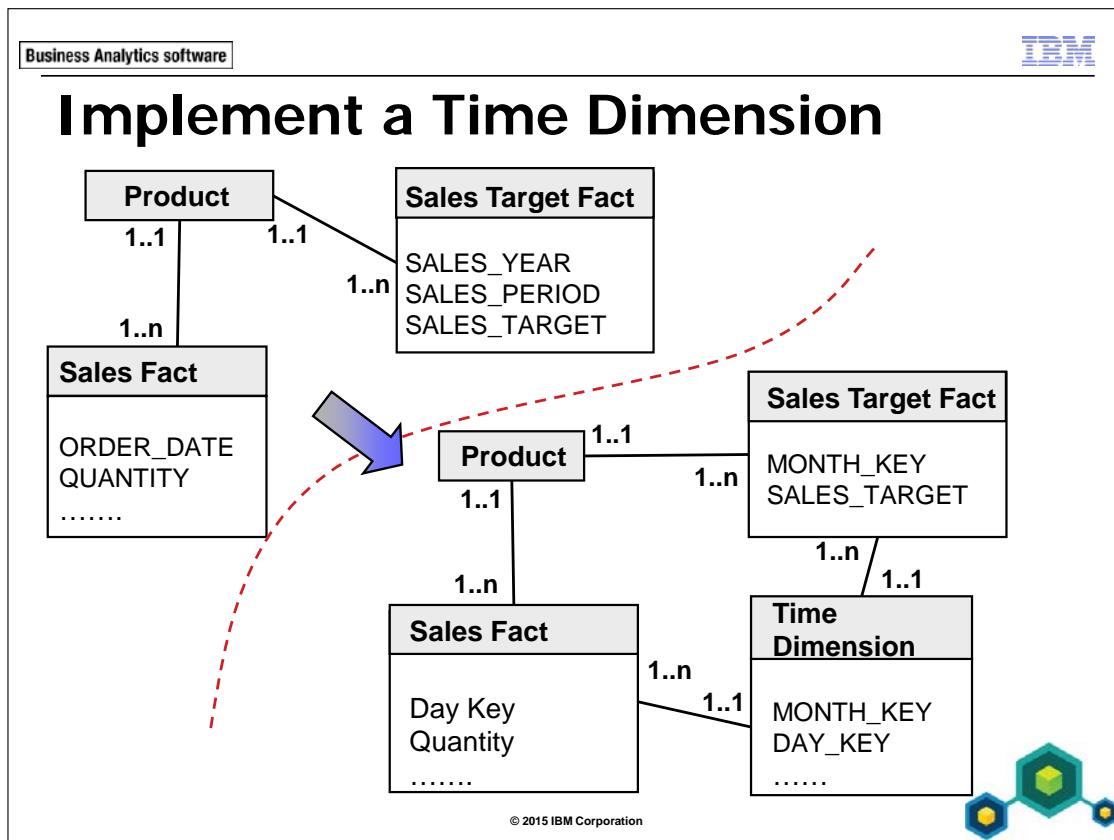
You could then sort in ascending order on the two new columns in order to match them to the SALES\_YEAR and SALES\_PERIOD columns from SALES\_TARGET.

However, this cannot be accomplished in Query Studio and therefore requires the modeler to create the calculations or restricts these types of calculations to Report Studio authors. The results also do not properly stitch the two fact queries because the items used to compare the data do not come from a conformed source.

## 6. Close IBM Cognos Workspace Advanced, without saving the report.

### Results:

**You identified the issues involved with trying to create a time-based report without a time dimension. Without a conformed time dimension, you cannot achieve a proper stitch query, and will require calculations to match date fields from different sources.**



A time dimension provides a conformed dimension that supports time-based queries across facts, either at the same or different levels of granularity. The current model has various date formats in the fact tables (such as ORDER\_DATE, SALES\_PERIOD) that make time-based queries between facts difficult. Time dimensions also allow for easy rollups, for example, day-based facts can be rolled up to the month level.

You can create a time dimension by using a table with all applicable dates for a time period, ensuring that it includes all required keys, date derivatives, and formats. There are several SQL scripts available on the internet to build time dimensions.

You can import the time dimension into Framework Manager and create relationships as necessary. If the fact query subjects do not contain the keys required to relate to the time dimension, speak to the database administrator to see if they can be incorporated.

You can also create time keys in Framework Manager using calculations, but this may reduce performance since you are filtering on non-indexed fields.

## Demo 2: Implement a Time Dimension

### Purpose:

In order to ensure time-based queries return predictable results, you will import and implement a time dimension.

Components: Framework Manager, Query Studio, Report Studio

Project: GO Operational

Package: GO Operational

### Task 1. Import Time Dimension metadata.

1. In **Framework Manager**, expand **Foundation Objects View**, right-click **gosales**, and then click **Run Metadata Wizard**.
2. Ensure that **Data Sources** is selected, and then click **Next**.
3. Ensure that **GOSALES** is selected, and then click **Next**.
4. Expand **GOSALES > Tables**, and then select **TIME\_DIMENSION**.
5. Click **Next**, and then click **Import**.
6. Click **Finish**.

### Task 2. Test and adjust TIME\_DIMENSION.

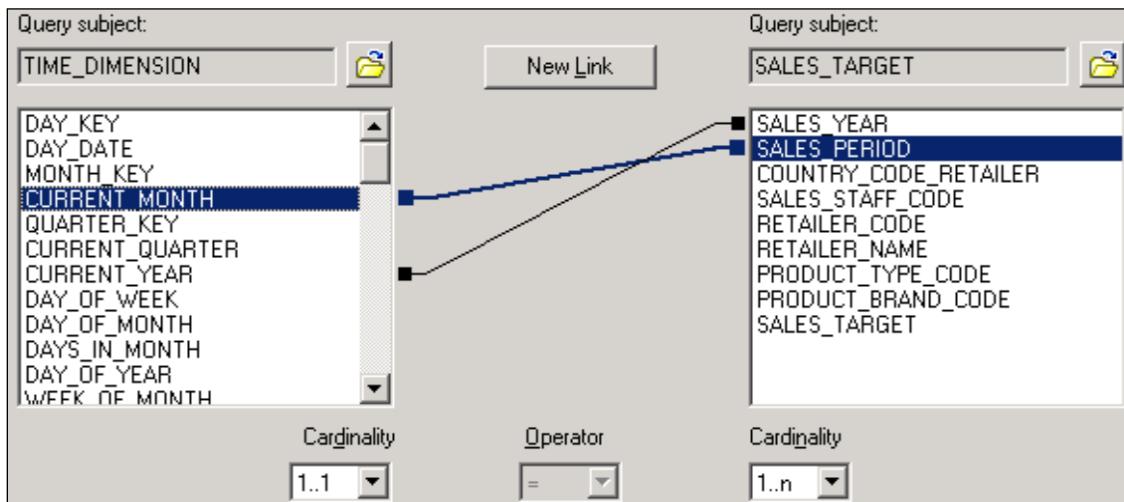
1. Drag **TIME\_DIMENSION** below the **Staff Region (alias)** query subject to organize it alphabetically.
2. Right-click **TIME\_DIMENSION**, and click **Launch Context Explorer**, and then click **Show Related Objects**.  
  
TIME\_DIMENSION is a generic query subject with no relationships to other query subjects.
3. Close **Context Explorer**.

4. Right-click **TIME\_DIMENSION**, click **Test**, and then click **Test Sample**. This query subject is simply a calendar breakdown, giving the time over a certain range in multiple formats. It can be used to join two fact tables with different date formats. In this case, **TIME\_DIMENSION**'s **DAY\_DATE** matches the format of Sales Fact's **ORDER\_DATE** and **TIME\_DIMENSION**'s **CURRENT\_YEAR** and **CURRENT\_MONTH** match **SALES\_TARGET**'s **SALES\_YEAR** and **SALES\_PERIOD**.
5. Click **Close**.
6. Expand **TIME\_DIMENSION**, click **CURRENT\_YEAR**, and then, in the **Properties** pane beside **Format**, click **<Click to edit>**.  
Note: If your Properties pane is not visible, then from the View menu click Properties.
7. Under **Format type**, select **Number**, set **Use Thousands Separator** to **No**, and then click **OK**.
8. In the **Project Viewer**, select all of the **TIME\_DIMENSION** query items that are identified as facts, and then change the **Usage** properties from **Fact** to **Attribute**.

### Task 3. Build relationships between TIME\_DIMENSION and facts.

1. In the **Project Viewer** pane, in the **Foundation Objects View**, create the following relationships:
  - **TIME\_DIMENSION** (DAY\_DATE, 1..1) to **Sales Fact** (ORDER\_DATE, 1..n)
  - **TIME\_DIMENSION** (DAY\_DATE, 1..1) to **Returns Fact** (RETURN\_DATE, 1..n)
2. Create the following relationship, but do not close the Relationship dialog box:
  - **TIME\_DIMENSION** (CURRENT\_YEAR, 1..1) to **SALES\_TARGET** (SALES\_YEAR, 1..n)

3. Click **New Link** to add a second segment to this two-segment relationship:
  - **TIME\_DIMENSION** (CURRENT\_MONTH, 1..1) to **SALES TARGET** (SALES\_PERIOD, 1..n)



Note the Expression below the linkages:

**TIME\_DIMENSION.CURRENT\_YEAR =  
SALES\_TARGET.SALES\_YEAR AND  
TIME\_DIMENSION.CURRENT\_MONTH =  
SALES\_TARGET.SALES\_PERIOD**

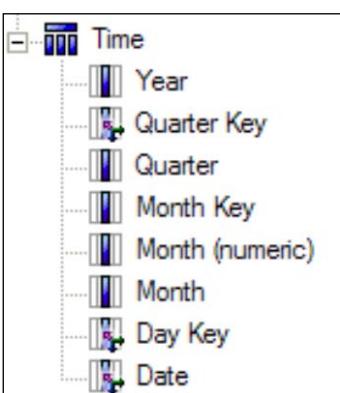
This relationship now consists of two values in TIME\_DIMENSION that match time period values in SALES\_TARGET. The fields used from the TIME\_DIMENSION table, in this scenario, are not indexed in the database. This may affect performance. The ideal situation would be to have a month key in the SALES\_TARGET table in the database. This would allow you to create your relationship based on one common indexed key in each table. If this is not an option, you should consider asking the database administrator to index the CURRENT\_MONTH and CURRENT\_YEAR fields in the TIME\_DIMENSION table if performance is an issue.

4. Click **OK**.

#### Task 4. Create the Time model query subject in the Consolidation View.

1. Right-click **Consolidation View**, point to **Create**, and then click **Query Subject**.
2. Name the query subject **Time**, and then click **OK**.

3. In the **Available Model Objects** pane, expand **Foundation Objects View > gosales > TIME\_DIMENSION**.
4. Add the following query items by double-clicking them:
  - **CURRENT\_YEAR**
  - **QUARTER\_KEY**
  - **CURRENT\_QUARTER**
  - **MONTH\_KEY**
  - **CURRENT\_MONTH**
  - **MONTH\_EN**
  - **DAY\_KEY**
  - **DAY\_DATE**
5. Click **OK**, and then rename the items as follows:
  - **Year**
  - **Quarter Key**
  - **Quarter**
  - **Month Key**
  - **Month (numeric)**
  - **Month**
  - **Day Key**
  - **Date**
6. Ensure that the items are organized as shown below:



7. Drag the **Time** query subject above the **Model Filters** folder.
8. Save the project.

This material is meant for IBM Academic Initiative use only. NOT FOR RESALE

## Task 5. Prepare the model for multilingual data.

1. In **Foundation Objects View > gosales > Reusable Objects**, right-click **Model Calculations**, point to **Create** and click **Calculation**.
2. In the **Name** box, type **Month**.
3. In the **Available Components** box, navigate to **Foundation Objects View > gosales > TIME\_DIMENSION** and drag the **MONTH\_EN** data item into the **Expression definition** box.
4. In the **Expression definition** box, add '#' in front of **[gosales]** and replace **EN]** with '**+ \$Language\_lookup{\$runLocale} + ]'**'#.

The completed expression should read as follows:

```
#'[gosales].[TIME_DIMENSION].[MONTH_'][$Language_lookup{$runLocale}+]#'
```

5. Click **OK**, and then save the project.

## Task 6. Publish and test package.

1. Publish the **GO Operational** package.
2. In **IBM Cognos Connection**, launch **Query Studio**, and then select the **GO Operational** package.
3. Create a new report with the following items from **Consolidation View**:

| Query Subject     | Query Item                            |
|-------------------|---------------------------------------|
| Time              | <b>Year</b><br><b>Month (numeric)</b> |
| Sales Target Fact | <b>Sales Target</b>                   |

A section of the results appear as follows:

| Year | Month (numeric) | Sales Target  |
|------|-----------------|---------------|
| 2010 | 1               | 1,786,471,100 |
| 2010 | 2               | 1,898,274,000 |
| 2010 | 3               | 2,161,992,700 |

Monthly sales targets are in the billions, which is too high. Double counting is occurring because there is not enough information specified on the time dimension to roll up values correctly at the month level. In the next module, you will learn how to specify determinants for the underlying **TIME\_DIMENSION** query subject to resolve this issue.

- Save the query as **Test Time Dimension** to the default location.

Note: It is important that you save this query, as it will be used in a subsequent demo.

You will now test a multi-fact query using the Time dimension as a conformed dimension.

- Create a new report with the following items from **Consolidation View**:

| Query Subject | Query Item                            |
|---------------|---------------------------------------|
| Time          | <b>Year</b><br><b>Month (Numeric)</b> |
| Sales Fact    | <b>Revenue</b>                        |
| Returns Fact  | <b>Return Quantity</b>                |

A section of the results appear as follows:

| Year | Month (numeric) | Revenue         | Return Quantity |
|------|-----------------|-----------------|-----------------|
| 2010 | 1               | \$72,741,622.65 | 596             |
| 2010 | 2               | \$73,242,843.99 | 8,129           |
| 2010 | 3               | \$75,720,238.67 | 10,341          |
| 2010 | 4               | \$65,278,358.76 | 17,898          |
| 2010 | 5               | \$74,695,218.83 | 27,693          |
| 2010 | 6               | \$82,169,806.98 | 43,562          |

This multi-fact query was accomplished using Time as a conformed dimension to stitch the two fact queries together.

To verify this, you will open the report in Report Studio to view the SQL.

- Under **Menu**, click **Manage File**, and then click **Open in Report Studio**.
- In **Report Studio**, from the **Tools** menu, click **Show Generated SQL/MDX**.

8. From the list, select **IBM Cognos SQL**.

The results appear as follows:

Generated SQL/MDX:

IBM Cognos SQL

```

select
 coalesce(D2.Year1,D3.Year1) as Year1,
 coalesce(D2.Month__numeric_,D3.Month__numeric_) as Month__numeric_,
 D2.Revenue as Revenue,
 D3.Return_Quantity as Return_Quantity,
 XSUM(D2.Revenue) as Revenue5,
 XSUM(D3.Return_Quantity) as Return_Quantity6
from
 (select
 TIME_DIMENSION.CURRENT_YEAR as Year1,
 TIME_DIMENSION.CURRENT_MONTH as Month__numeric_,
 XSUM(Sales_Fact.Revenue for TIME_DIMENSION.CURRENT_YEAR,TIME_DIMENSION.CURRENT_MONTH) as Revenue
 from
 GOSALES..GOSALES.TIME_DIMENSION TIME_DIMENSION,
 (select
 ORDER_HEADER.ORDER_DATE as ORDER_DATE,
 (ORDER_DETAILS.QUANTITY * ORDER_DETAILS.UNIT_SALE_PRICE) as Revenue
 from
 GOSALES..GOSALES.ORDER_HEADER ORDER_HEADER,
 GOSALES..GOSALES.ORDER_DETAILS ORDER_DETAILS
 where
 (ORDER_HEADER.ORDER_NUMBER = ORDER_DETAILS.ORDER_NUMBER)
) Sales_Fact
 where
 (TIME_DIMENSION.DAY_DATE = Sales_Fact.ORDER_DATE)
 group by
 TIME_DIMENSION.CURRENT_YEAR,
 TIME_DIMENSION.CURRENT_MONTH
) D2
full outer join
 (select
 TIME_DIMENSION.CURRENT_YEAR as Year1,
 TIME_DIMENSION.CURRENT_MONTH as Month__numeric_,
 XSUM(D2.Return_Quantity for TIME_DIMENSION.CURRENT_YEAR,TIME_DIMENSION.CURRENT_MONTH) as Return_Quantity6
 from
 GOSALES..GOSALES.TIME_DIMENSION TIME_DIMENSION,
 (select
 ORDER_HEADER.ORDER_DATE as ORDER_DATE,
 (ORDER_DETAILS.QUANTITY * ORDER_DETAILS.UNIT_SALE_PRICE) as Revenue
 from
 GOSALES..GOSALES.ORDER_HEADER ORDER_HEADER,
 GOSALES..GOSALES.ORDER_DETAILS ORDER_DETAILS
 where
 (ORDER_HEADER.ORDER_NUMBER = ORDER_DETAILS.ORDER_NUMBER)
) Sales_Fact
 where
 (TIME_DIMENSION.DAY_DATE = Sales_Fact.ORDER_DATE)
 group by
 TIME_DIMENSION.CURRENT_YEAR,
 TIME_DIMENSION.CURRENT_MONTH
) D3

```

If you scroll through the SQL, you will see that there are two queries merged together using a full outer join.

9. Click **Close**, close **Report Studio** and **Query Studio**, and then close your browser.

## Results:

**By implementing a Time dimension, you can now easily query across fact query subjects. You also discovered that Sales Target values are not correct, and will correct this later in the modeling process by using determinants.**

**Business Analytics software**

**IBM**

## Resolve Multiple Ambiguous Joins

- Use role-playing time dimensions to resolve multiple ambiguous joins.

The diagram shows a central 'Sales Fact' box connected by three red lines to three separate boxes labeled 'Time', 'Time (Ship)', and 'Time (Close)'. Handwritten-style blue text labels above the lines read 'Order Date = Day Key', 'Ship Date = Day Key', and 'Close Date = Day Key' respectively. A small copyright notice '© 2015 IBM Corporation' is at the bottom left, and a decorative graphic of colored hexagons is at the bottom right.

A table with multiple valid relationships between itself and another table presents a reporting trap. Sales Fact, for example, has three possible ways of joining to the Time dimension. To ensure that queries use the correct relationship, create two additional aliases for the Time dimension and assign each a role: one links to Sales Fact by Order Date, one by Ship Date, and one by Close Date.

## Demo 3: Resolve Multiple Ambiguous Joins

### Purpose:

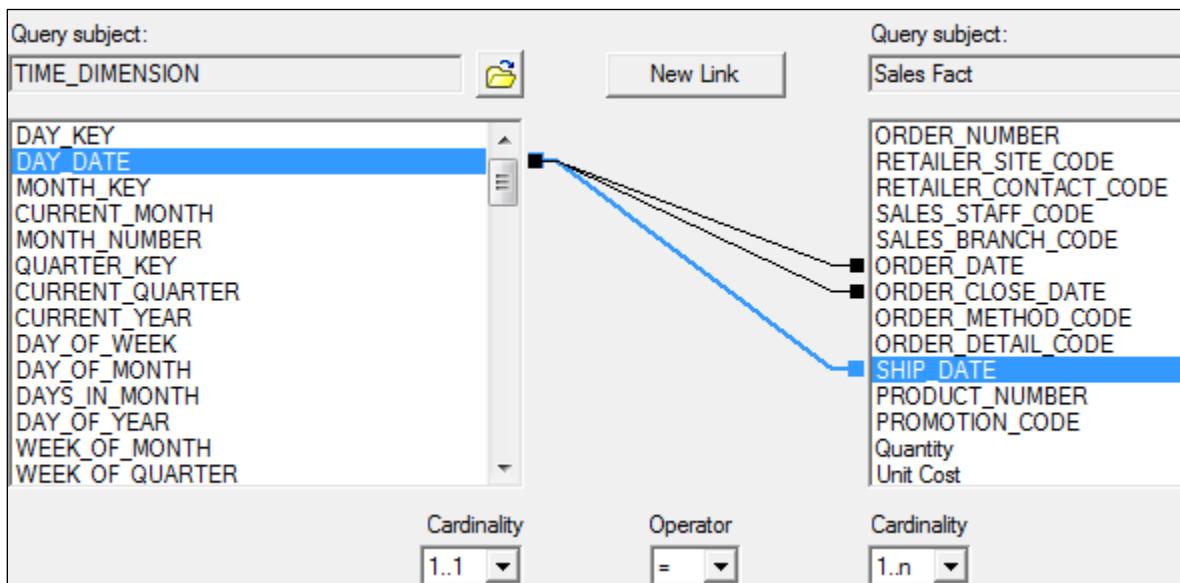
Report authors require queries based on shipping and closing dates as well as the standard order date. However, while analyzing your model design, you see that this causes multiple relationships between Sales Fact and Time. This means that proper join may not be performed at run time. You will resolve this ambiguity by creating role-playing dimensions for each relationship.

Component: Framework Manager

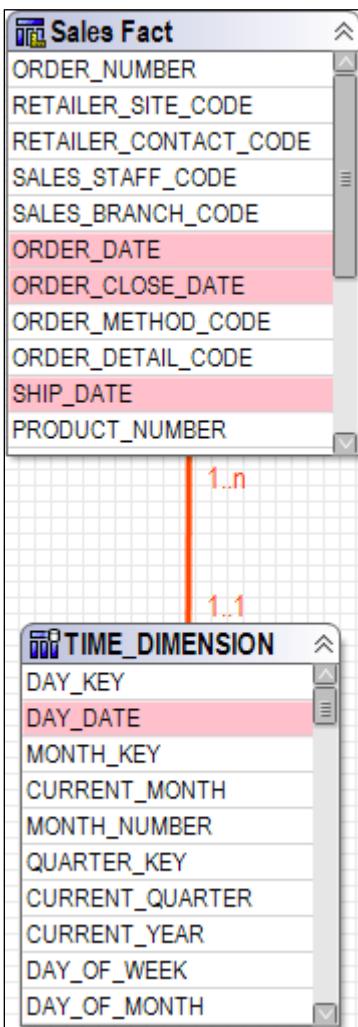
Project: GO Operational

### Task 1. Use Model Advisor to Identify the Issue.

1. In Foundation Objects View > gosales, right-click Sales Fact, click Launch Context Explorer, and then click Show Related Objects.
2. Double-click the relationship between Sales Fact and **TIME\_DIMENSION**. Note that DAY\_DATE is joined to ORDER\_DATE. There should also be a relationship to ORDER\_CLOSE\_DATE and SHIP\_DATE.
3. Create two new links between **TIME\_DIMENSION** and Sales Fact:
  - DAY\_DATE (1..1) to ORDER\_CLOSE\_DATE (1..n)
  - DAY\_DATE (1..1) to SHIP\_DATE (1..n)



4. Click **OK** to close the Relationship Definition dialog box.
5. In **Context Explorer**, click the relationship between **Sales Fact** and **TIME\_DIMENSION**.



Notice the multiple relationships between Sales Fact and TIME\_DIMENSION. You cannot use the relationships in this state and get expected results. There is not enough information for IBM Cognos to select the correct relationship for different queries. Next, you will implement role playing time dimensions to handle ship dates and close dates.

6. Double-click the relationship, and delete the two new relationships on **SHIP\_DATE** and **ORDER\_CLOSE\_DATE**.
7. Click **OK**, and then close the **Context Explorer**.

## Task 2. Create a role-playing dimension (alias).

1. In **Foundation Objects View > gosales**, create a new model query subject called **Time (Close)**, and then click **OK**.
2. In the **Available Model Objects** pane, expand **Foundation Objects View > gosales > TIME\_DIMENSION**, and then double-click the following query items to add them to the definition:
  - **DAY\_DATE**
  - **CURRENT\_YEAR**
  - **QUARTER\_KEY**
  - **CURRENT\_QUARTER**
  - **MONTH\_KEY**
  - **CURRENT\_MONTH**
  - **MONTH\_EN**
  - **DAY\_KEY**
3. Click **OK**, and then drag **Time (Close)** below **TIME\_DIMENSION**.
4. In **Time (Close)**, rename the data items to:
  - **Close Date**
  - **Close Year**
  - **Close Quarter Key**
  - **Close Quarter**
  - **Close Month Key**
  - **Close Month (numeric)**
  - **Close Month**
  - **Close Day Key**
5. Create a relationship from **Time (Close) (Close Date, 1..1)** to **Sales Fact (ORDER\_CLOSE\_DATE, 1..n)**.

6. Right-click **Time (Close)**, and click **Edit > Copy**, and then right-click the **gosales** namespace, and click **Edit > Paste**.
7. Rename the copy to **Time (Ship)**, and then rename the data items to:
  - **Ship Date**
  - **Ship Year**
  - **Ship Quarter Key**
  - **Ship Quarter**
  - **Ship Month Key**
  - **Ship Month (numeric)**
  - **Ship Month**
  - **Ship Day Key**
8. Move **Time (Ship)** below **Time (Close)**, and creating the following relationship:  
**Time (Ship)** (Ship Date, 1..1) to **Sales Fact** (SHIP\_DATE, 1..n)
9. In the **Time (Close)** query subject, double-click **Close Month**, and delete the expression in the **Expression definition** box.
10. From **gosales > Reusable Objects > Model Calculations**, add the **Month** calculation, and then click **OK**.
11. In **Time (Ship)**, double-click **Ship Month**, replace the existing expression with the **Month** calculation, and then click **OK**.
12. In the **Project Viewer**, select and **Test** the following query items with **Auto Sum** disabled:

| Query Subject  | Query Item          |
|----------------|---------------------|
| Sales Fact     | <b>ORDER_NUMBER</b> |
| TIME_DIMENSION | <b>DAY_DATE</b>     |
| Time (Ship)    | <b>Ship Date</b>    |
| Time (Close)   | <b>Close Date</b>   |
| Sales Fact     | <b>Quantity</b>     |

A section of the results appear as follows:

| ORDER_N | DAY_DATE                 | Ship Date                | Close Date               | Quantity |
|---------|--------------------------|--------------------------|--------------------------|----------|
| 100001  | Jan 12, 2010 12:00:00 AM | Feb 17, 2010 12:00:00 AM | Feb 17, 2010 12:00:00 AM | 256      |
| 100001  | Jan 12, 2010 12:00:00 AM | Feb 17, 2010 12:00:00 AM | Feb 17, 2010 12:00:00 AM | 92       |
| 100002  | Jan 12, 2010 12:00:00 AM | Jan 19, 2010 12:00:00 AM | Jan 19, 2010 12:00:00 AM | 162      |
| 100002  | Jan 12, 2010 12:00:00 AM | Jan 19, 2010 12:00:00 AM | Jan 19, 2010 12:00:00 AM | 172      |
| 100002  | Jan 12, 2010 12:00:00 AM | Jan 19, 2010 12:00:00 AM | Jan 19, 2010 12:00:00 AM | 74       |
| 100002  | Jan 12, 2010 12:00:00 AM | Jan 19, 2010 12:00:00 AM | Jan 19, 2010 12:00:00 AM | 90       |
| 100002  | Jan 12, 2010 12:00:00 AM | Jan 19, 2010 12:00:00 AM | Jan 19, 2010 12:00:00 AM | 422      |
| 100003  | Jan 12, 2010 12:00:00 AM | Jan 20, 2010 12:00:00 AM | Jan 20, 2010 12:00:00 AM | 3252     |
| 100003  | Jan 12, 2010 12:00:00 AM | Jan 20, 2010 12:00:00 AM | Jan 20, 2010 12:00:00 AM | 1107     |
| 100003  | Jan 12, 2010 12:00:00 AM | Jan 20, 2010 12:00:00 AM | Jan 20, 2010 12:00:00 AM | 88       |
| 100003  | Jan 12, 2010 12:00:00 AM | Jan 20, 2010 12:00:00 AM | Jan 20, 2010 12:00:00 AM | 19       |
| 100004  | Jan 12, 2010 12:00:00 AM | Jan 21, 2010 12:00:00 AM | Jan 21, 2010 12:00:00 AM | 354      |

In this test results the close dates are the same day as the ship date. However, it is possible that some orders will have a later close date, since larger orders do not always ship all products on the same day.

13. Click **Close**, save the project and then close Framework Manager.

### Results:

**You saw that multiple relationships between Sales Fact and Time meant that a proper join might not be performed at run time. You resolved this ambiguity by creating role-playing dimensions for the SHIP\_DATE and ORDER\_CLOSE\_DATE relationships.**

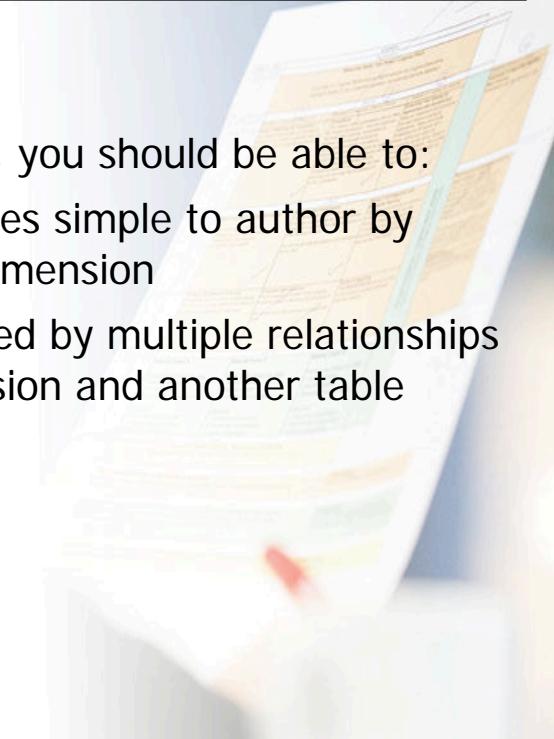
Business Analytics software

IBM

## Summary

- At the end of this module, you should be able to:
  - make time-based queries simple to author by implementing a time dimension
  - resolve confusion caused by multiple relationships between a time dimension and another table

© 2015 IBM Corporation



This material is meant for IBM Academic Initiative use only. NOT FOR RESALE

10-22

© 2003, 2015, IBM Corporation

This guide contains proprietary information which is protected by copyright. No part of this document may be photocopied, reproduced, or translated into another language without a legal license agreement from IBM Corporation.

This material is meant for IBM Academic Initiative use only. NOT FOR RESALE

This information was developed for products and services offered in the U.S.A.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not grant you any license to these patents. You can send license inquiries, in writing, to:

*IBM Director of Licensing  
IBM Corporation  
North Castle Drive  
Armonk, NY 10504-1785  
U.S.A.*

**The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law:** INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

If you are viewing this information softcopy, the photographs and color illustrations may not appear.