

WHERESCAPE RED TERADATA TUTORIALS 6.8.5.0

www.wherescape.com

WhereScape RED Teradata Tutorials

The software described in this book is furnished under a license agreement and may be used only in accordance with the terms of agreement.

Copyright Notice

Copyright © 2002-2016 WhereScape Software Limited. All rights reserved. This document may be redistributed in its entirety and in this electronic or printed form only without permission; all other uses of this document and the information it contains require the explicit written permission of WhereScape Software limited.

Due to continued product development this information may change without notice. WhereScape Software Limited does not warrant that this document is error-free.

Trademarks

WhereScape and WhereScape RED are trademarks or registered trademarks of WhereScape Software Limited. Other brands or product names are trademarks or registered trademarks of their respective companies.

WhereScape USA, Inc

1915 NW AmberGlen Parkway Suite 400, Beaverton Oregon 97006 United States T: 503-466-3979 F: 503-466-3978

WhereScape Limited

P.O.Box 56569, Auckland 1446 12-16 Tapora Street Quay Park Auckland 1010, New Zealand T: +64-9-358-5678 T: US toll free 1-877-237-3980 F: +64-9-358-5679

WhereScape Europe

Reading Enterprise Centre Earley Gate, Whiteknights Rd Reading RG6 6BU United Kingdom T: +44-118-914-4509 F: +44-118-914-4508

WhereScape Asia Pte. Ltd

300 Tampines Avenue 5 #09-02 Singapore 529653 T: +65-6679-5728

CONTENTS

Tutorial Overview	1
The First Step	2
Logging In	3
Tutorial 1 - Basic Star Schema Fact Table	4
1.1 Purpose and Roadmap	5
1.1.1 Prerequisite Actions	8
1.2 Adding an Odbc Data Source	9
1.3 Creating a Connection	13
1.4 Loading Source Tables	20
1.4.1 Loading Source Tables using Schemas	26
1.5 Building Dimensions	31
1.6 Creating Dimension Views	42
1.7 Defining the Staging Table	48
1.8 Including Dimension Links	56
1.9 Creating a Fact Table	66
1.10 Switching to Diagrammatic View	72
1.11 Producing Documentation	77

i

CHAPTER 1 TUTORIAL OVERVIEW

In Tutorial 1 we will create a basic Star Schema Fact Table. Additional Tutorials may be available on the WhereScape website.

IN THIS CHAPTER

The First Step	2
Logging In	3

THE FIRST STEP

The first step

To get started you need to follow the steps in the **WhereScape Installation and Administration Guide** to create the required environment. The basic steps in this process are:

The basic steps in this process are:

- **1** Install the WhereScape product suite.
- 2 Create a database user under which to load the WhereScape metadata repository.
- **3** Install the WhereScape metadata repository.

You are now ready to *log on* (see "*Logging In*" on page 3) to the repository you have created.

LOGGING IN

Having completed the first step, and using WhereScape RED, you can now log on to the repository you have created .

To log in:

1 Click WhereScape RED from the Start menu. The Access Control screen displays. See sample screen below:

		x
		REPOSITORY LOGIN
		DATABASE
WhoroScopo®	Data Source:	TD_14_00 ¥
where Scape	Logon Method:	Teradata Wallet 🗸 🗸
RED	Database Login ID:	dssdemo
	TD Wallet String:	Your Teradata Wallet ID
Version 6.8.4.2 by WhereScape Software Limited		METADATA REPOSITORY
Licensed to RED Documentation	RED Database:	dssdemo
For WhereScape employee use only	RED User Name:	WhereScape Documentation
		HELP CANCEL CONNECT

- 2 The **Data Source** is the ODBC connection for your database. This connection must have been established prior to logon. Refer to the Installation and Administrator Guide if the connection doesn't exist.
- **3** The **Logon Method** is the Method chosen to login. Select either DB User/Password or Teradata Wallet method from the drop-down menu.
- **4** The **Database Login ID** is the User Name where the user has been granted access to the user under which the metadata repository has been loaded.
- **5** The **TD Wallet String** is the string replacing the user name and password for your connection. Teradata TD Wallet is a Teradata product part of the TTU (Tools and Utilities). Refer to Teradata documentation if you don't have a TD Wallet created already.
- 6 The **RED Database** is the User Name for the metadata repository.
- 7 The **RED User Name** is the name that will be associated with any procedures, tables, etc, and scheduled jobs that are created within RED. Normally this would be your full name.
- 8 Click **OK**. The Builder screen displays.

CHAPTER 2 TUTORIAL 1 - BASIC STAR SCHEMA FACT TABLE

IN THIS CHAPTER

1.1 Purpose and Roadmap	. 5
1.2 Adding an Odbc Data Source	.9
1.3 Creating a Connection	. 13
1.4 Loading Source Tables	. 20
1.5 Building Dimensions	. 31
1.6 Creating Dimension Views	. 42
1.7 Defining the Staging Table	. 48
1.8 Including Dimension Links	. 56
1.9 Creating a Fact Table	. 66
1.10 Switching to Diagrammatic View	. 72
1.11 Producing Documentation	. 77

1.1 PURPOSE AND ROADMAP

Purpose

This tutorial is designed to introduce you to basic development using WhereScape RED. At the end of the tutorial you will have built a simple dimensional analysis area of a data warehouse, with identity column surrogate keys.

Note: Surrogate keys using identity columns are not always the recommended approach in Teradata. This first tutorial uses them for simplicity. It is important to note that WhereScape RED can be used with and without surrogate keys and when used with surrogate keys, they can be derived from identity columns or using any other user defined approach.

The tutorial will build the star-schema shown below. This star-schema comprises a central fact table, fact_sales_detail, joined to four dimension tables. Data will be loaded from tables in another Teradata database.

In the process of creating this star-schema you will learn to create load, stage, fact and dimension tables. You will also see how data flows from the data source, through the different tables enroute to its fact or dimension table destination.



Tutorial Roadmap

This tutorial works through a number of steps. These steps and the relevant section within the chapter are summarized below to assist in guiding you through the tutorial.

Step in Tutorial	Section
Create a new odbc data source for the RED tutorial source system	Adding an Odbc Data Source
Create a new RED connection object for the RED tutorial source system	Creating a Connection
Create and load the load tables for	Loading Source Tables
• Customer	
• Product	
Order_line	
• Order_header	

Step in Tutorial	Section
Create (and update from load tables) the following dimension tables	Building Dimensions
 Dim_customer Dim product 	
Create dimensions for Dim_order_date Dim_ship_date These are views on the dim_date table	Creating Dimension Views
 Create the stage_sales_detail table Create stage table using columns from load_order_line and load_order_header Specify join condition Include links to the following dimensions (dim_customer, dim_product, dim_order_date, dim_ship_date) 	Defining the Staging Table Including Dimension Links
Create the fact_sales_detail table	Creating a Fact Table
View the WhereScape generated documentation	Switching to Diagrammatic View Producing Documentation

1.1.1 PREREQUISITE ACTIONS

Before commencing this tutorial, please perform the following tasks:

- 1 Install RED metadata
- 2 Install Tutorial data

For more information, refer to the WhereScape Installation and Administration Guide.

1.2 ADDING AN ODBC DATA SOURCE

We need to add an Odbc DSN called TutorialTDAT, which will be used to connect to the Tutorial Source data.

1 Open WhereScape Administrator and select PC Setup > Validate / Add Odbc Source.



2 Right-click in the middle work pane and click **Add Odbc entry**. **WhereScape Administrator** will open ODBC Administrator. Choose the **System DSN** tab and click **Add** to add a System Data Source.

Name	Platform	Driver		Add
EDW_Prod_	32-bit 64-bit	leradata Netezza SQI		Remove
NZSQL	32-bit	NetezzaSQL		
ORCL	64-bit	Oracle in OraClient12Home1		Configure
ORCL	32-bit	Oracle in OraClient12Home1_32bit		-
ORCL_Sales	32-bit	Oracle in OraClient12Home1_32bit		
ORCL_Sales2	32-bit	Oracle in OraClient12Home1_32bit		
ORCL_Tut	32-bit	Microsoft ODBC for Oracle		
Red 1	64-bit	SQL Server Native Client 11.0	~	
C		>		
An ODBC Sys A System data	tem data source st source is visible to	ores information about how to connect all users of this computer, including	ct to the i NT servi	ndicated data provider. ces.

3 Select a driver for your Data Source, choose **Teradata** and click **Finish**.

Create New Data Source	×
Select a driver for which you want to set up a dat Name Netezza SQL Oracle in OraClient 12Home 1_32bit Postgre SQL ANSI Postgre SQL Unicode SQL Server SQL Server SQL Server Native Client 11.0 Teradata	ta source.
 < Back Finish	Cancel

- **4** Enter a Data Source **Name** of **Teradata_Tutorial**. As the Teradata Server is environment specific, enter the name of your Teradata Server.
- **5** Enter either User name and Password or your TD Wallet String. **Click Ok**.

ODBC	Driver S	etup for Teradata Data	base ×
Data Source			OK
Name:	Teradata	a_Tutorial	UK I
Description:			Cancel
			Help
Teradata Serv	er Info		
Name or IP address	TD_14_0	00	\sim
Authentication	1		
Use Integra	ated Security	/	
Mechanism:		×	
Parameter:			Change
Username:	tutorial		
OPassword			
Teradata W	/allet String	type your TD Wallet String	
		\$tdwallet(type your TD Wallet	: String)
Optional			
Default Datab	ase:		
Account String]:		Options >>
Session Charact	er Set:		
ASCII			~

6 Click **OK** to close.

Name		Platform	Driver			^	Add
TD_14_00 td_dbc		32-bit 32-bit	Teradata Teradata				Remove
TERA1410		32-bit	Teradata				
Teradata_Sales		32-bit	Teradata		- 1		Configure
Teradata_Sales2		32-bit	Teradata				
Teradata_Tutorial		32-bit	Teradata		- 1		
test1		32-bit	SQL Server				
TrainingTD		32-bit	Teradata				
tsales		32-bit	SQL Server Native	Client 11.0		×	
<					>		
An ODBC Sy A System dat	stem data ta source i	source st is visible to	ores information abo all users of this cor	ut how to co nputer, inclu	onnec ding l	t to the i NT servi	ndicated data provider. ces.

7 In the left pane, click on **Validate** / **Add Odbc Source** again to refresh the middle pane and display the new ODBC Source Teradata_Tutorial.

1.3 CREATING A CONNECTION

In order to populate the metadata repository, a connection needs to be created for the tutorial source system.

This section describes how to create this new connection.

TIP: In order to use the drag and drop features there must always be a connection to the data warehouse itself.

1 In the left pane of the RED Builder, click on and highlight the **Connection object group**. (This selects the object group to be worked on.)



• From the right mouse menu select New Object.

- **2** A dialog box displays with the Object Type defaulted to Connection.
 - Name your connection **Tutorial(OLTP)** and click **ADD**.

	Add a New Metadata Object ×
Define the Type Specific informat	and Name of the New Object. ion for each object type is defined in subsequent screens.
Object Type:	Connection 🗸
Object Name:	Tutorial(OLTP)
	ADD Cancel

- **3** The Connection Definition dialog displays next. Complete the details as below and click **OK**:
 - The **ODBC source** is Teradata_Tutorial.
 - The **Database ID (SID)** can be left blank.
 - The **Database Link ID** can be left blank.
 - The **Extract User ID** and **password** are the user name and password for a user who has select access to the tutorial database (or user) where the source tables reside.
 - If the **ODBC User Default** is set to Teradata Wallet, enter the User ID who has select access to the tutorial database and the relevant Teradata Wallet String.
 - The Administrator User ID and password can be left blank for the tutorial.
 - The **New Table Default Load Type** enables you to set the default load type at connection level for ODBC and Database Connections. Set to 'Database link load'.
 - Leave Data Type Mapping set as 'Default'.

es	General Connection Name		
	Connection Name		
		Tutorial(OLTP)	
	Connection Type	Database	~
	Database Type	(local)	~
	ODBC Data Source Name (DSN)	Teradata_Tutorial	~
	Data Warehouse Connection Indicator	False	~
	▲ Source System		
	Database ID	TD_14_00	
	Database Link Name		
	Database Credentials		
	Extract User ID		
	Extract User Password		
	Administrator User ID		
	Administrator User Password		
	Teradata Wallet User ID	tutorial	
	Teradata Wallet String	type your TD Wallet String	
	ODBC User Default	Teradata Wallet	\sim
	▲ Other		
	New Table Default Load Type	Database link load	\sim
	SSIS Connection String		
	Data Type Mapping Set	(Default)	~

> > > > >
> > >
> > >
> >
>
~
~
~
*

4 To confirm that you have connected to the system correctly, right-click on the new connection object in the left pane and select **Browse Connection**.

Properties New Version	
Create a SQL Window Connection (Administrator User) Create a SQL Window Connection (Extract User)	
Browse Connection	
Telnet Window	
Create Remote View Procedure	
Create Database Link	
Delete	
Documentation	×
Projects	×
Impact	×

5 Ensure the **Schema** is set to the database (or user) where tutorial tables have been created and click **OK**.

Note: If the Extract User ID is selected, the Schema will have the user ID and the password encrypted. If the TD Wallet has been selected instead, the Browse Connection Window will show the TD Wallet String by default on the password window and the TD Wallet String check-box will be ticked.

	List Source Tables Connection
Connection:	Tutorial(OLTP)
User ID:	tutorial
Password:	*****
Filter	Password is TDWallet string.
Schema:	tutorial
Name:	(None)
	Object Types Image: Tables Image: Views Image: System Objects
Group:	(All)
Project:	(All) V
Data Type Mapp Refresh Current	oing Set: (Default) OK Cancel

	List Source Tables Connection	×
Connection:	Tutorial(OLTP)	
User ID:	tutorial	
Password:	☐ D Wallet String will be shown here by default)	
Filter	Password is TDWallet string.	
Schema:	tutorial	
Name:	(None)	
	Object Types Image: Tables Image: Views System Objects	
Group:	(All) V	
Project:	(All) V	
Data Type Mapp Refresh Current	ing Set: (Default) V OK Canc	el

6 A third pane on the right displays the tables contained in the tutorial source system:

Tutorial (OLTP) 🛛 👻 🕂 🗙
 tutorial continent country customer model_customer model_forecast order_header order_line prod_group prod_line prod_subgroup product state
Data Warehouse Source Browser

You have created a database connection for the source system **Tutorial (OLTP)** and are now ready to proceed to the next step - *Loading Source Tables* (see "*1.4 Loading Source Tables*" on page 20)

1.4 LOADING SOURCE TABLES

In this step you will load data from the tutorial source system into load tables in the data warehouse.

Dragging and dropping from the source system (using the previously defined connection) will create the metadata.

You will be prompted to create and load the tables which will create the physical tables in the data warehouse and then load the data.

TIP: In order to utilize the drag and drop features, ensure that your source system is displayed in the right pane.

1 Double click on the **Load Table object group** on the Object Tree in the left pane. The first column heading in the middle pane should read **Load**.

Load				
Load Name	Connection	Schema	Database Link	Database
<				>

2 Ensure the Object Tree is expanded in the right pane.



- **3** Click on **customer** and drag this table into the middle pane, placing it anywhere in the pane. A dialog box displays with the name of the object defaulted to **load_customer**.
- If you want to place this table on a different schema, use the Target Location drop-down list to select the right schema for your table. To see more details about placing tables in different schemas, see further instructions on the next topic *Loading Source Tables using schemas*.

	Add a New Metadata Object	×
Define the Type and	Name of the New Object.	
Specific information for	or each object type is defined in subsequent screens.	
Object Type:	Load Table	~
Object Name:	load_customer	
Target Location:	(local)	*
Data Type Mapping:	(default)	*
	ADD Cance	el

4 Click **Add** and the following table definition will display:

		Lo	ad load_customer		
Properties	Load Table Name:	load_custo	mer		
Storage Override Create DDI	Unique Short Name: (maximum 22 characters)	load_custo	mer		
Source	Description:				
Notes					
	Connection:	Tutorial(OL	TP)	~	
	Load Type:	Database li	nk load	~	
	Database Link:				
	Script Name:	(None)		\vee	
	Pre-Load Action:	Truncate		~	
		<			>
	Post Load Procedure:	(None)		¥	
	Timestamps Metadata Structure Cha 2014-10-30 00:45:53.8%	anged: 20000	Database Created: 2014-10-30 00:46:23.970000		Database Altered: 2014-10-30 00:46:23.970000
					OK Cancel Help

Note: For the purposes of this tutorial, all the necessary details have been created automatically, so click **OK**. See the Loading Data chapter for explanations of the load table fields and options.

5 A dialog box displays showing that the load table load_customer has been defined and asks if you want to create and load the table. Click Create and Load.

Create Database Table	×
Load load customer has been defined	
Create Create and Load 🔻	Close
Create Create and Load 🔻	Close

6 Click **Yes** to define a Primary Index.



7 Type code as the on the Primary Index Columns field. This will create the physical tables in the data warehouse and load the data.Click OK on the Storage screen

Click OK on the	Storage screen.
------------------------	-----------------

Properties Storage Dveride Create DDL Source Database disademo v Notes Primary Index Non-Unique Primary Index (NUP!) v Primary Index Columns code			Load load_customer		×
Storage Override Create DDL Source Notes Primary Index Name Ioad_customer_idx_PR Primary Index Columns Code Other Optional CREATE Clause	Properties	2			
Override Create DDL. Source Notes Detabase dssdemo Primary Index Columns code MultiSet True Fallback False Other Optional CREATE Clause Primary Index Columns Columns Columns Columns Columns Columns Optional CREATE Clause	Storage	▲ Storage			
Source Temp Database dsdemo v Noles Primary Index Non-Unique Primary Index (NUPI) v Primary Index Columns code MultiSet True v Fallback False v Data Block Size v v Enable Free Space False v Other Optional CREATE Clause v Primary Index Columns code Optional CREATE Clause Primary Index Columns Columns of the Primary Index. Dialog Opening Value: code, address	Override Create DDL	Database	dssdemo		¥
Johnson Primary Index Notes Primary Index Name load_customer_idx_PR Primary Index Columns Code Columns of the Primary Index Primary Index Columns Columns of the Primary Index. Primary Index Columns Columns of the Primary Index. Detailed address <	Source	Temp Database	dssdemo		¥
Notes Primary Index Columns code Primary Index Columns code MultiSet True Fallback False Data Block Size False Image: Columns Columns Optional CREATE Clause Primary Index Columns Columns of the Primary Index. Dialog Opening Value: code, address OK Cancel Help	Jource	Primary Index	Non-Unique Primary Index (NUPI)		×
Primary Index Columns code MultiSet True Pallback False Data Block Size v Enable Free Space False Other v Optional CREATE Clause v Primary Index Columns v Columns of the Primary Index. Dialog Opening Value: code, address DK Cancel	Notes	Primary Index Name	load_customer_idx_PR		
MultiSet True Fallback False Data Block Size Enable Free Space False Other Optional CREATE Clause Primary Index Columns Columns of the Primary Index. Dialog Opening Value: code, address OK Cancel Help		Primary Index Columns	code		
Fallback False Data Block Size Enable Free Space False Optional CREATE Clause Optional CREATE Clause Primary Index Columns Columns of the Primary Index. Dialog Opening Value: code, address OK Cancel Help		MultiSet	True		~
Data Block Size Enable Free Space False		Fallback	False		~
Enable Free Space False Other Optional CREATE Clause Primary Index Columns Columns of the Primary Index. Dialog Opening Value: code,address OK Cancel Help		Data Block Size			~
Other Optional CREATE Clause ··· Primary Index Columns Columns of the Primary Index. Dialog Opening Value: code,address OK Cancel Help		Enable Free Space	False		~
Optional CREATE Clause Primary Index Columns Columns of the Primary Index. Dialog Opening Value: code,address OK Cancel Help		▲ Other			
Primary Index Columns Columns of the Primary Index. Dialog Opening Value: code,address		Optional CREATE Clause			
Primary Index Columns Columns of the Primary Index. Dialog Opening Value: code,address OK Cancel					
Primary Index Columns Columns of the Primary Index. Dialog Opening Value: code,address OK Cancel					
Primary Index Columns Columns of the Primary Index. Dialog Opening Value: code,address OK Cancel					
Primary Index Columns Columns of the Primary Index. Dialog Opening Value: code,address OK Cancel					
Primary Index Columns Columns of the Primary Index. Dialog Opening Value: code, address OK Cancel					
Primary Index Columns Columns of the Primary Index. Dialog Opening Value: code,address OK Cancel					
Primary Index Columns Columns of the Primary Index. Dialog Opening Value: code,address OK Cancel					
Primary Index Columns Columns of the Primary Index. Dialog Opening Value: code,address OK Cancel					
Primary Index Columns Columns of the Primary Index. Dialog Opening Value: code,address OK Cancel					
Primary Index Columns Columns of the Primary Index. Dialog Opening Value: code,address OK Cancel Help					
Primary Index Columns Columns of the Primary Index. Dialog Opening Value: code,address OK Cancel Help					
Primary Index Columns Columns of the Primary Index. Dialog Opening Value: code,address OK Cancel Help					
Primary Index Columns Columns of the Primary Index. Dialog Opening Value: code,address OK Cancel Help					
Primary Index Columns Columns of the Primary Index. Dialog Opening Value: code,address OK Cancel					
Columns of the Primary Index. Dialog Opening Value: code,address OK Cancel Help		Primary Index Columns			
Dialog Opening Value: code,address OK Cancel Help		Columns of the Primary Index.			
OK Cancel Help		Dialog Opening Value: code,addre	55		
OK Cancel Help					
				OK C	ancel Help

8 The physical tables will now be created in the data warehouse and the data will be loaded. The results will be posted in the results pane.

Object Message Image: Image	Re	sults	→ ‡ ×
Image: Second	٩	Object	Message
 Ioad_customer CREATE MULTISET TABLE dssdemo.load_customer, No Fallback (code decimal(6) TITLE 'code', name varchar(45) TITLE 'name', address varchar(40) TITLE 'address', city varchar(30) TITLE 'city', state varchar(2) 'state') PRIMARY INDEX Ioad_customer_idx_PR (code) ; Ioad_customer Ioad_customer Ioad_customer Ioad_customer Ioad_customer Table dssdemo.load_customer Ioad_customer Ioad_Costomer Ioad_customer Ioad_customer Ioad_Costomer Ioad_Costomer Ioad_Costomer Ioad_Costomer Ioad_Costomer	0	load_customer	Table dssdemo.load_customer created.
 load_customer table comment added. load_customer 0 column comments added. load_customer Table dssdemo.load_customer truncated. load_customer Insert into dssdemo.load_customer load_customer Select load_customer customer.code, customer.name, customer.address, customer.city, customer.state load_customer From tutorial.customer load_customer ; load_customer ; load_customer 6 rows inserted. 	0	load_customer 	CREATE MULTISET TABLE dssdemo.load_customer , No Fallback (code decimal(6) TITLE 'code', name varchar(45) TITLE 'name', address varchar(40) TITLE 'address', city varchar(30) TITLE 'city', state varchar(2) TITLE 'state') PRIMARY INDEX load_customer_idx_PR (code) ;
 Ioad_customer Ioad_customer Ioad_customer Table dssdemo.load_customer truncated. Ioad_customer Insert into dssdemo.load_customer Ioad_customer Select Ioad_customer customer.code, customer.name, customer.address, customer.city, customer.state Ioad_customer Ioad_customer From tutorial.customer Ioad_customer 	0	 load_customer 	table comment added.
Image: Select Image: Select Image: Select	0	 load_customer 	0 column comments added.
Image: Select Image: Select Image: Select	0	 load_customer 	Table dssdemo.load_customer truncated.
 Ioad_customer Ioad_customer Ioad_customer customer.code, customer.name, customer.address, customer.city, customer.state Ioad_customer From tutorial.customer Ioad_customer 	0	 load_customer 	Insert into dssdemo.load_customer
 load_customer customer.code, customer.name, customer.address, customer.city, customer.state load_customer 	0	 load_customer 	Select
 load_customer 	0	 load_customer 	customer.code, customer.name, customer.address, customer.city, customer.state
 load_customer ; load_customer 6 rows inserted. 	0	 load_customer 	From tutorial.customer
Icad_customer 6 rows inserted.	0	 load_customer 	;
load sustamer Load Completed Normally, 6 rows inserted	0	 load_customer 	6 rows inserted.
United Completed Normally, o rows inserted.	0	load_customer	Load Completed Normally. 6 rows inserted.

- 9 Notice that the Load Table object group in the left pane now has a dependent/child.
- **10** Repeat this process (steps 1 7) for the source tables **product**, **order_header**, and **order_line**. When selecting the Primary Index for each table, choose:
 - **code** for product table
 - order_number for order_header table
 - **order_number** for order_line table

- **TIP**: Remember to double click on the left pane Load Table object group between loading each of the source tables to ensure that you are reassigning the target, rather than adding to the columns in the middle pane.

11 Your screen should look something like this:

File Edit View Browse Backup	Jobs Doc Reports Vali	date Tools Window	Help			
🗄 🗋 🚲 🖒 😤 🤣 Web Links 👻 🕘	🔒 📇 🕓 Scheduler 🥔	• 🛷 • 📮 🛛 🕱 🔌	Import 📕 👿	🥔 🌾 📮 🗄 🖗		
<mark>ё Builder</mark> ও Scheduler 🛛 🚑 Diagrar	n					-
Development 👻 म 🗙	Load				Tutorial(OLTP)	₩ ₽ ×
All Objects	Load Name	Connection	Schema	Database Link	a 🗖 👌	
Connection	load_customer	Tutorial(OLTP)	tutorial			
Load	load_order_header	Tutorial(OLTP)	tutorial		📄 % 🛪 💥 🗱 🎞 Y (0
🐅 Stage	Ioad_order_line	Tutorial(OLTP)	tutorial		a 📋 tutorial	
🗱 Data Store	load_product	Tutorial(OLTP)	tutorial		continent	
T Normalized					country	
Dimension					model customer	
🙀 Fact					model_forecast	
🍬 Join Index					T order_header	
ø Aggregate					T order_line	
I Data					prod_group	
etro						
Olap Cube					product	
Export					T state	
Procedure						
Host Script	<			>		
index	D					
	Results			₩×		
	🚷 Object 🛛	Message		^		
	🎯 🗕 load_order_line S	Select				
	🎯 🗕 load_order_line 🛛	order_line.order_number, o	order_line.order_li	ne_no,orde		
	🎯 🗕 load_order_line 🛛	order_line.unit_sale_price,	order_line.quantit	y,order_lin		
	☑ _ load_order_line	From tutorial.order_line				
		21 rows inserted.				
	☑ └ load_order_line I	Load Completed Normally	7. 21 rows inserted	l.		
		Create of Load dssdemo.lo successfully.	oad_product com	pleted 🗸 🗸		
	Results Reports				Data Warehouse Source Brow	ser
Deedu	Middle Densyl and	Development (decideme	ويراجعون المراجع والمراجع	Descusor Test		INC .

You are now ready to proceed to the next step - *Building Dimensions* (see "1.5 *Building Dimensions*" on page 31)

1.4.1 LOADING SOURCE TABLES USING SCHEMAS

Tables can be placed in different target locations on a connection level which enables simpler handling of deployments and database storage changes. To place tables in different target locations users should follow the below steps:

Overview

- Ensure the relevant target databases exist in Teradata. Create any databases that do not exist.
- Set the **Enable Targets for setting object location** option in the <PRODUCT> Repository Identification options.
- Add one or more target locations to the Data Warehouse connection in <PRODUCT> for each database you intend to use.
- Configure the Data Warehouse connection in <PRODUCT> to browse all required databases by default.
- Set the default target locations for **new tables** in the <PRODUCT> **Table Location** options.
- When defining a new table in <PRODUCT>, check and ensure the correct target is set on the **Storage** tab.
- 1 After logging in to <COMPANY> <PRODUCT>, make sure the **Enable Targets for setting object location** option is set in the **Tools->Options->Repository Identification** settings.

B	Optio	ns	_ 🗆 🗙
Repository Identification			
Repository Privacy Settings	A Repository		
Object Types	Repository Name	Development	
 Bobal Naming Conventions DSS Tables and Columns 	Repository Type	Development	2
Check-Out and Check-In	Data Warehouse Database	dssdemo	
> Code Generation	Enable Targets for setting object location		
Storage	Enable Pargets for setting object location		
Metadata Versioning			
Documentation			
Other			
			Old Constant Utals

- **2** Add one or more target locations to the Data Warehouse connection in <PRODUCT> for each target database you want to use:
 - Click the **Add** button to add the required target locations for this connection.

entes			
tes	Connection Name	DataWarehouse	^
	Connection Type	Database	~
	Database Type	Teradata	*
	ODBC Data Source Name (DSN)	RED_REPOSITORY	~
	Data Warehouse Connection Indicator	\checkmark	
	Source System		
	Database ID	TD_14_00	
	Database Link Name		
	Database Credentials		
	Extract User ID		
	Extract User Password		
	Administrator User ID		
	Administrator User Password		
	Teradata Wallet User ID	RED_REPOSITORY	
	Teradata Wallet String	your TD Wallet String	
	ODBC User Default	Extract User	~
	▲ Other		
	Default Schema for Browsing	RED_REPOSITORY	
	New Table Default Load Type	Database link load	~
	SSIS Connection String (OLEDB)	Provider=TDOLEDB.1;Password=\$PASSWORD\$;P	Persist Sec
	SSIS Use Column Names		
	Data Type Mapping Set	(Default)	~
	Default Transform Function Set	(Default)	~
	When Connection is an OLAP Data Source		
	Target Table Location		
	Add new Target Location	Add	

3 Give the new **Target Database** a name and then enter the target's database. It is best to set the target name to the same name as the target database.

Add Target				
Add a new target for the connection				
Target Name:	RED_STAGE			
Target Database:	RED_STAGE			
Temp Database:	Blank if not required or for default temporary database			
	OK Cance			

4 Expand the target locations to change target database location colors or to delete target databases.

RED_STAGE		
Name	RED_STAGE	
Database	RED_STAGE	
List Color	197; 21; 23	
Temp Database		
Delete Target Location	Delete	
		V

5 Still in the **DataWarehouse** connection, add the new databases to the **Default Schema for Browsing** field separated by commas.

Connection Da	taWarehouse	
operties		
otes deneral		^
Connection Name	DataWarehouse	
Connection Type	Database	¥
Database Type	Teradata	~
ODBC Data Source Name (DSN)	RED_REPOSITORY	¥
Data Warehouse Connection Indicator		
▲ Source System		
Database ID	TD_14_00	
Database Link Name		
Database Credentials		
Extract User ID		
Extract User Password		
Administrator User ID		
Administrator User Password		
Teradata Wallet User ID	RED_REPOSITORY	
Teradata Wallet String	your TD Wallet String	
ODBC User Default	Extract User	¥
▲ Other		
Default Schema for Browsing	RED_REPOSITORY, RED_STAGE, RED_MODEL, RED_BAS	E_V
New Table Default Load Type	Database link load	¥
SSIS Connection String (OLEDB)	Provider=TDOLEDB.1;Password=\$PASSWORD\$;Persist Se	c
SSIS Use Column Names		
Data Type Mapping Set	(Default)	V
Default Transform Function Set	(Default)	~
When Connection is an OLAP Data Source		
OLAP Connection String		
Connection Provider/Driver		
Default Schema for Browsing Optional comma-delimited list of schema for browse	er pane filter.	

NOTE: While browsing this connection, <PRODUCT> will then display a list with all the target databases and their associated objects on the right-hand browser pane.

- **6** You are also able to set the default target location for **New Tables** in **Tools/Options**. This default target location is only applied when a new table is created, not for existing tables.
 - Select between the **Set Target** option to set a default target location for new tables (or to to use the table's storage) or
 - Same as Source to place new tables in the same database as their source.

II.	Opt	ions	_ 🗆 🗙
Repository Identification	Ai		
Repository Privacy Settings	· 2♥		
> Object Types	Load lable Default larget		
Global Naming Conventions	Target Action	Set Target	~
DSS Tables and Columns	Default Target	(local)	~
Check-Out and Check-In	Stage Table Default Target		
 Code Generation A Storage 	Target Action	Same as Source	~
Target Location	Dimension Table Default Target		
Table Storage	Target Action	Same as Source	~
Default Optional CREATE Clause	▲ Fact Table Default Target		
Index Type	Target Action	Same as Source	~
Metadata Versioning	Aggregate Table Default Target		
Documentation	Target Action	Same as Source	~
Other	Join Index Table Default Target		
	Target Action	Same as Source	~
	Data Store Table Default Target		
	Target Action	Same as Source	~
	EDW 2NE Table Default Target	Same as source	
	Target Action	Samo as Source	
	A View Default Target	Jame as Jource	•
	a view belaute larget	6 C	
	larget Action	Same as Source	~
Prev Next			OK Cancel Help

Follow the usual steps for creating objects by using the drag and drop functionality.
 As you are defining a new table in <PRODUCT>, ensure the correct target location options are set in the **Target Location** options before creating the table in the database.
 When using drag and drop, it is also possible to change the target database location of a

particular object as you create it.

After a table has been created, it is also possible to change its target location on the **Storage** tab of the table's Properties screen.

NOTE: When upgrading from a <PRODUCT> version previous to 6.8.2.0 and moving existing objects to a target location, all procedures that reference those objects will need to be rebuilt. Any **FROM** clauses will also need to be manually regenerated in order for the table references to be updated to the new [TABLEOWNER] form.

1.5 BUILDING DIMENSIONS

The necessary source tables have been loaded into the data warehouse. Now the dimensions of the data warehouse can be built.

When building dimensions you will be prompted as to how you would like the dimension managed. WhereScape RED generates code for normal, slowly changing and date ranged dimensions.

You will also be prompted for the business (or natural) key of the dimension. This is needed so WhereScape RED knows when to add new dimensional records.

1 Double-click on the **Dimension Table object group** to expand the dimension folder in the left pane.



2 Right-click on **dim_date** and select **Execute Update Procedure**.

Properties	
Storage	
Display Columns	
Display Indexes	
Display Data	
Query via Excel	
Add Column	
Add Index	
Regenerate Indexes	
Hierarchies	•
Change Column(s)	
Validate Against the Database	
Update Comments	
Gather Statistics	•
Version Control	•
Create (ReCreate)	
Truncate	
Delete Metadata and Drop Table	
Execute Update Procedure	
Execute Custom Procedure	
Process Table via Scheduler	
Execute Custom Procedure via Scheduler	
Documentation	•
Projects	•
Impact	•
Code	•

3 Right-click on **dim_date** and select **Execute Custom Procedure**.

Properties Storage	
Display Columns Display Indexes Display Data Query via Excel	
Add Column Add Index Regenerate Indexes	
Hierarchies	►
Change Column(s) Validate Against the Database Update Comments Gather Statistics	•
Version Control	×
Create (ReCreate) Truncate Delete Metadata and Drop Table	
Execute Update Procedure	
Execute Custom Procedure	
Process Table via Scheduler Execute Custom Procedure via Scheduler	
Documentation	+
Projects	•
Impact	+
Code	•
4 Right-click on the **DataWarehouse** connection in the left pane and select **Browse Connection**.

Properties New Version	
Create a SQL Window Connection (Administrator User)	
Create a SQL Window Connection (Extract User)	
Browse Connection	
Telnet Window	
Create Remote View Procedure	
Create Database Link	
Delete	
Documentation	•
Projects	•
Impact	•

5 The Source Tables Connection will display either your encrypted password or your TD Wallet string depending on the connection properties database credentials' option chosen. Click **OK**.

	List Source Tables Connection	×
Connection:	DataWarehouse 🗸	
User ID:	dssdemo	
Password:	TD Wallet String	
Filter	Password is TDWallet string.	
Schema:	dssdemo	
Name:	(None)	
	Object Types Image: Tables Image: Views System Objects	
Group:	(All) 🗸	
Project:	(All) V	
Data Type Mapp Refresh Current	oing Set: (Default)	el

6 This shows the data warehouse tables and views in the right pane.



Note: From this point on, all work is performed within the data warehouse.

7 Double click on the Dimension object group in the object tree in the left pane.The first column of the middle pane now reads Dimension.

🔡 Builder 🕓 Scheduler	🚊 Diagram					
Development	→ ᡎ ×	Dimension				45
All Objects		Dimension Name	Short Name	Dimension Type	Database	Update Procedure
Connection		😽 dim_date	dim_date	Time	dssdemo	update_dim_date
Load		😽 model_date	model_date	Time	dssdemo	update_model_date
辩 Stage						
🗱 Data Store						
T Normalized						
b 💏 Dimension						
翼 Fact						
🐚 Join Index						
🧳 Aggregate						
💢 View						
Retro						
Olap Cube						
Utap Dimension						
Export						
B Host Script						
		<				>

Note: You will see that some dimensions have already been created for you.

8 Click and drag the **load_customer** table from the data warehouse schema in the right pane into the middle pane.

A dialog box displays defaulting the name of the object to **dim_customer**. Click **ADD**.

	Add a New Metadata Object	×						
Define the Type and Name of the New Object.								
Specific information for each object type is defined in subsequent screens.								
Object Type:	Dimension	*						
Object Name:	dim_customer							
Target Location:	(local)	~						
	ADD Canc	el						

9 A Dimension Type dialog is displayed. Click **Normal**.

Dimension Type	×
Four methods are provided for managing dimensions. Please select the desired method.	0
 Normal. The dimension is updated based on a business key, with new records being added if required. All columns except the business key can change. 	Normal
Slowly changing. Changes in the values of selected columns result in new dimensional records being created. In all other respects the same as Type 1.	Slowly Changing
Previous data retained. The previous values of selected columns are stored in additional columns. In all other respects the same as Type 1.	Previous values
4. Date Ranged. The source system provides a date ranged business key. Similar to Type 2 except that we deal with the record as a whole and the dates are provided.	Date Ranged

10 A table definition displays with all the necessary defaults completed.

- Make one change Select **(Build Procedure...)** from the Update Procedure drop-down list box this will generate procedures to get surrogate (artificial) keys based on the business key and to update the dimension.
- Click **OK**.

*		Dimensi	ion dim_customer					x
Properties Storage Override Create DDL	Table Name: Unique Short Name: (maximum 22 characters)	dim_customer				Table Type:	Dimension	~
Language Mapping Purpose	Business Display Name (EUL):	dim_customer						
Concept Grain	Description:							^
Examples								~
Usage	Update Procedure:	(Build Procedur	e)	~	Rebuild			
Notes	Custom Procedure:	(None)		~				
	Get Key Function:	(None)		▼ Edit			Mnemonic (EUL):	
	Timestamps Metadata Structure Chang	ed [.]	Database Created:		Dati	ahase Altered		
	2014-10-30 03:58:06.1500	00	2014-10-30 03:58:18	020000	201	4-10-30 03:58:1	8.020000	
						ОК	Cancel H	elp

11 A dialog box displays confirming that the dimension table **dim_customer** has been defined and asking if you want to create and load the table. Click **Create and Load**.

Cr	eate Database Tab	le ×				
Dimension dim_customer has been defined.						
Create	ate and Load	Close				
		Close				

12 Click **Yes** to define a Primary Index.

	WhereScape RED						
?	The Primary Index has not been defined. Click Yes to define a Primary Index before creating the table. Click No to create the table as a NOPI table. Click Cancel to not create the table.						
	Yes No Cancel						

Type **code** as the Primary Index Column and click **OK** on the Storage screen.

88		Dimension dim_customer			×
Properties	₽				
Storage	▲ Storage				
Override Create DDL	Database	dssdemo			~
Language Mapping	Primary Index	Non-Unique Primary Index (NUPI)			~
	Primary Index Name	dim_customer_idx_PR			
Purpose	Primary Index Columns	code			
Concept	MultiSet	True			~
Grain	Fallback	False			· • ·
	Data Block Size				· • ·
Examples	Enable Free Space	False			· • ·
Usage	Other				
Notes	Optional CREATE Clause				
	Primary Index Columns				
	Columns of the Primary Index.				
	Dialog Opening Value: code, city				
			OK	Cancel	Help

14 Define the Business Key as below when the next dialog appears. The business (natural) key is the unique identifier for the dimensional record. Type Code and and click OK.

	dim_cust	omer Update Build Options.	
Information	<u>∰</u> .		
Processing			
Source	Business Key Columns	code	
	Parameters	-	
	Insert Zero Key Record	Irue	
	Include Initial Load Insert	False	
	 Batch Processing 		
	Process by Batch	False	
	Delete Processing		
	Delete Before Insert	No	
	Update Processing		
	Process Method	Insert/Update	
	Insert Method		
	 Include Insert Statement 	True	
	✓ Insert New Rows Only	True	
	New Row Identification Method	Join	
	▲ Update Method		
	Include Update Statement	True	
	Update Changed Rows Only	True	
	Change Row Identification Method	Join	
	Business Key Columns Columns that define the business key for updat	e processing. Required if include Update options.	

15 Notice that the results will be posted in the Results pane.

Re	sults		•	·д	×
٩	Object	t	Message		-
0	🖃 din	n_customer	Table dssdemo.dim_customer created.		
0	_	dim_customer	CREATE MULTISET TABLE dssdemo.dim_customer, No Fallback (dim_customer_key integer generated by default as identity (start with 1 increment by 1) TITLE 'dim customer key' NOT NULL, code decimal(6) TITLE 'code', name varchar(45) TITLE 'name', address varchar(40) TITLI 'address', city varchar(30) TITLE 'city', state varchar(2) TITLE 'state', dss_update_time timestam TITLE 'dss update time') PRIMARY INDEX dim_customer_idx_PR (code) ;	E	
0	-	dim_customer	table comment added.		
0	-	dim_customer	2 column comments added.		Ξ
0	-	dim_customer	Procedure Completed		
0	L	dim_customer	1 dim_customer updated. 6 records added. 0 records updated.		
					-
Re	ports	Results			

16 Repeat this same process (steps 7 through 16) for the load table **load_product**. The Primary Index and Business key will be **code**.

TIP: Remember to double-click on the left pane Dimension Table object group between loading each of the above dimension tables.

- **17** Double click **Dimension** in the left pane.
- **18** The WhereScape RED screen should look like this:

File Edit View Browse Backup	Jobs Doc Reports V	alidate Tools Wir	ndow Help		
E Vilder	🔒 📇 🕓 Scheduler 🧉	₽ ▼	🕹 🌛 Import 📋	🛛 🖉 🖗 🐐 🗐 🕅	
Development - $P \times$	n dim_product Columns			•	► DataWarehouse ► ₽ ×
■ Al Objects ■ Connection ■ Load ♥ Stage \$ Data Store ■ Normalized ■ Stage \$ Dimension \$ dim_customer \$ dim_date \$ dim_fordate \$ model_date \$ Join Index \$ Aggregate \$ View ■ Retro ④ Olap Cube	Column Name Column Name Colum	Display Name dim product key code description prod line prod group subgroup dss update time	Data Type integer gene decimal(6) varchar(64) varchar(24) varchar(24) varchar(24) timestamp	Source Table load_product load_product load_product load_product load_product	DataWarehouse A warehouse A model with a second seco
 i Olap Dimension i Export i Procedure i Host Script i Motex 	< Results Object / dim_product dim_product dim_product dim_product dim_product dim_product dim_product dim_product dim_product	Message COMMENT ON COI dssdemo.dim_produ time the row was up COMMENT ON COI dssdemo.dim_produ artificial key'; Procedure Complet 1 dim_product upd updated.	.UMN uct.dss_update_ti dated in the data .UMN uct.dim_product_ ed ated. 9 new recor	we IS 'Date and a warehouse.'; key IS 'Generated ds. 0 records	Data Warehouse Source Browser

You are now ready to proceed to the next step - *Creating Dimension Views* (see "1.6 *Creating Dimension Views*" on page 42)

1.6 CREATING DIMENSION VIEWS

A dimension view is a database view of a dimension table. It may be a full or partial view. It is typically used in such cases as date dimensions where multiple date dimensions exist for one fact table.

In this step you will create dimension views from an existing dimension. In many cases dimension views are built as part of the end user layer, but creating them in the data warehouse means they are available regardless of the end user tools used. This process is essentially the same as creating a dimension, but you are creating a view of an existing dimension, in this instance, dim_date.

1 Double-click on **View** in the left pane. The first column heading in the middle pane should read **View**.

View			4.
View Name	Short Name	View Type	Get Key Procedure
<			>

2 Click and drag **dim_date** from the right pane into the middle pane.



3 The dialog box that displays defaults the object type to a dimension view, and names the view **view_date**.

We want to create two dimension views from the same source, dim_date, so we need to change this dimension view name to one that is more meaningful, specifically **dim_order_date.**

	Add a New Metadata Object	×
Define the Type and Specific information	Name of the New Object. for each object type is defined in subsequent screens.	
Object Type:	Dimension View	~
Object Name:	dim_order_date	
Target Location:	(local)	*
	ADD Cano	;el

4 Make this change and click **ADD**.

Make the following changes to the Column Definition dialog and click **OK**.

View Column Definition						
The column names for the view being created can be modified by filling in the following form. If the Default button is pressed nothing will be changed.						
Remove Column Prefix:	>	Add Column Prefix:				
Remove Business Display Prefix:	>	Add Business Display Prefix: o				
- Change Column Names for Specific Columns Old Column Name:		New Column Name:				
dim_date_key	>	dim_order_date_key				
calendar_date	>	order_date				
	>					
	>					
	>					
		OK Default				

6 On the dim_order_date Properties dialog, change the View Type to **Dimension View** and click **OK**.

4		Vie	w dim_order_da	te			
Properties Storage View Aliases Purpose	View Name: Unique Short Name: (maximum 22 characters) Business Disnlau Name (FLII):	dim_order_d	late			View Type:	Dimension View 🗸
Concept Grain Examples	Description:						~
Notes	Update Procedure: Custom Procedure: Distinct Data Select: From/Where or Where Clause:	(None)		v v	Rebuild	Regenerate	
	Table Locking Mode: Timestamps Metadata Structure Cha 2014-10-30 01:02:55.85	LOCK ROW	/ FOR ACCESS Database Crea 2014-10-30 01	✓ ated: :03:50.330000	Dat 201	abase Altered: 4-10-30 01:03:	Mnemonic (EUL):
						ОК	Cancel Help

7 Click Create View.

Create Database View	×
View dim _order_date has been defined	
Create View	Close

8 Repeat steps 1-7 to create the dimension view **dim_ship_date**. On the column Definition dialog, make the following changes:

The column names for the view being crea he Default button is pressed nothing will be	ited can be i e changed.	nodified by filling in the following form. If	•
Remove Column Prefix:		Add Column Prefix:	
	>	s_	
Baarawa Buuinawa Dinalay Bratin		Add Business Display Prefix:	
hemove business Display Prefix:		Add Dusiness Display Frenk.	
- Change Column Names for Specific Colu	>	s	
- Change Column Names for Specific Colu Old Column Name: dim_date_key	mns >	New Column Name: dim_ship_date_key	
- Change Column Names for Specific Colu Old Column Name: dim_date_key calendar_date	mns >	New Column Name: dim_ship_date_key ship_date	
- Change Column Names for Specific Colu Old Column Name: dim_date_key calendar_date	mns>	New Column Name: dim_ship_date_key ship_date	
- Change Column Names for Specific Colu Old Column Name: dim_date_key calendar_date	mns>	New Column Name: dim_ship_date_key ship_date	

9 Click in the right pane and press F5 to refresh the Data Warehouse table view in the right pane. Your screen should look something like this:

File Edit View Browse Backup	Jobs Doc Reports Va	ilidate Tools Wir	ndow Help			
🗄 📄 🙏 🏷 📇 🤣 Web Links 👻 🚱	🚡 🚉 🕓 Scheduler 🥪	• • 🛷 • 🚽 🛙 🧭 🤇	🔧 🌛 Import	🗏 🕱 🖗 🛒 🚆	15	
🔡 Builder 💧 Scheduler 🛛 🛃 Diagrar	n					•
Development • 4 ×	dim_ship_date Columns				- 🕼	DataWarehouse
🔺 🦳 All Objects	Column Name	Display Name	Data Type	Source Table	^	(2 T N
Connection	💢 dim_ship_date_key	dim ship date	integer	dim_date		
Load	💢 ship_date	ship date	date	dim_date		🗎 🕈 % 💥 🗱 🎞 🛛 🏹 🛞
🐅 Stage	💢 s_cal_day_in_week	scal day in week	varchar(3)	dim_date		🔺 🧻 dssdemo
🗱 Data Store	💢 s_cal_day_in_week_no	scal day in wee	integer	dim_date		dim_customer
T Normalized	💢 s_cal_day_in_month	scal day in mo	integer	dim_date		dim_date
a 器 Dimension	💢 s_cal_day_in_year	scal day in year	integer	dim_date		dim_order_date
😽 dim_customer	💢 s_cal_week_in_year	scal week in year	integer	dim_date		T dim ship date
😽 dim_date	💢 s_cal_month_no	scal month no	integer	dim_date		T dss_batch
😽 dim_product	💢 s_cal_month	scal month	integer	dim_date		<pre>dss_current_day_details</pre>
😽 model_date	💢 s_cal_month_name	scal month na	varchar(7)	dim_date		<pre>midss_fact_table</pre>
act 🙀	💢 s_cal_quarter_no	scal quarter no	integer	dim_date		dss_parameter
🐛 Join Index	💢 s_cal_quarter	scal quarter	integer	dim_date		load customer
Aggregate	💢 s_cal_year	scal year	integer	dim_date		load order header
A 💢 View	💢 s_financial_date	sfinancial date	date	dim_date		T load_order_line
aim_order_date	💢 s_fin_day_in_week	sfin day in week	varchar(3)	dim_date		load_product
Retro	💢 s_fin_day_in_week_no	sfin day in wee	integer	dim_date	~	T model_date
 Olan Cube 	<				>	
Olap Dimension	Degulta			_	лх	
Export	Results			·	Ť ^	
Procedure	🕙 Object 🛛 🛆	Message			^	
Host Script		Create of View dssde successfully.	emo.dim_ship_o	date completed		
	dim_ship_date	CREATE VIEW dssde dim_ship_date_key, ship_date, s_cal_day_in_week, s_cal_day_in_week, r_s_cal_day_in_wear, s_cal_day_in_year, s_cal_week,in_year,	mo.dim_ship_d 10, ,	late(~	
	Results Reports					Data Warehouse Source Browser

You are now ready to proceed to the next step - *Defining the Staging Table* (see "1.7 *Defining the Staging Table*" on page 48).

1.7 DEFINING THE STAGING TABLE

In this step you will create a stage table from two load tables. A stage table is used to build the format of the fact table, and generally contains changed or new data that will be added to the fact table. As stage tables contain dimensional keys, they should be defined after the dimensions.

NOTE: The source of data for the stage table will be the load tables load_order_line and load_order_header.

1 Double click on the Stage Table object group in the object tree in the left pane to create a stage table target.

Гhe first	column	heading	in the	middle	pane	reads	Stage.
		0			±		

Stage					
Stage Name	Short Name	Stage Type	Database		
<			>		

2 Click and drag the load_order_header table from the right hand pane data warehouse schema. Drop it in the middle pane.

A dialog box displays defaulting the name of the object to stage_order_header. To make it a more meaningful name, change the name of the object to **stage_sales_detail** and click **ADD**.

Add a New Metadata Object						
Define the Type and Name of the New Object.						
Specific information for each object type is defined in subsequent screens.						
Object Type:	Stage Table	*				
Object Name:	stage_sales_detail					
Target Location:	(local)	۷				
	ADD Cance	!				

3 A table definition displays with all the required defaults completed. Click **OK**

*		Stage st	age_sales_detail					×
Properties Storage Override Create DDL	Table Name: Unique Short Name: (maximum 22 characters)	stage_sales_deta	1) i)			Table Type:	Stage	*
Notes								
	Description:							~
	Update Procedure: Custom Procedure:	(None)		~	Rebuild	Regenerate		
	Timestamps Metadata Structure Chang 2014-12-03 03:26:57.450	ged: 000	Database Created:		Data	base Altered:		_
						ок	Cancel	Help

4 The Stage Table object in the left pane now has a dependent/child.



5 To add the remaining information from the second load table, click on stage_sales_detail in the left pane. Next drop load_order_line from the right pane and into the middle pane. A message is displayed with options to create a 'New Table' or to 'Add Columns'. Click **Add Columns**.

	Adding Table to Existing Stage	×
You are about The following o	to add all columns in the table load_order_line into this table (stage_sales_detail). iptions are available:	
New Table Add Columns Cancel	 will initiate a new table dialog. This table will not be affected will continue to add the columns to this table. No Action. 	
	New Table Add Columns Cancel	

6 WhereScape RED detects duplicate columns. As both load_order_header and load_order_line have the order number field, the following is displayed.

Click **Skip** to exclude the second instance of order_number - this combines data from two load tables (load_order_header and load_order_line) into one stage table.

Duplicate Column Name ×							
A column 'order_number' already exists. To add this column with a new name, edit and select OK To not add this column select Skip To add duplicate name select OK without editing (duplicates will need to be removed before the table can be created)							
order_number							
OK Skip							

- 7 In the middle pane, double-click on order_date to bring up the Properties screen for that column and change the Data Type to date.
- Repeat this same process for ship_date.

Properties Transformation		stage_sales_detail order_date order date	Pa Pa
Transformation	General Table Name Column Name Column Title Column Description Physical Definition Column Order Data Type Nutlicher Alleged	stage_sales_detail order_date order date 20	Pa Pa
	Table Name Column Name Column Title Column Description Physical Definition Column Order Data Type Null/Chan Allenged	stage_sales_detail order_date order date 20	20 20
	Column Name Column Title Column Description Physical Definition Column Order Data Type Nutlicities Allowed	order_date order date 20	년 년
	Column Title Column Description Physical Definition Column Order Data Type Null/Chan Allenged	order date	
	Column Description Physical Definition Column Order Data Type Null/Column Column	20	
	Physical Definition Column Order Data Type Notified Allowed	20	
	Column Order Data Type	20	
	Data Type		
	Null Values Alleured	date	
	Null values Allowed	True	¥
	Default Value		
	Character Set		
	Format		
	Character Comparison/Sorting		~
	Compress	False	×
	Meta Definition		
	Numeric	False	~
	Additive	False	~
	Attribute	True	\checkmark
	Business Key	False	×
	Key Type (0,A,B,C)		
	Source Details		
	Source Table	load_order_header	~
	Source Column	order_date	\checkmark
	Transformation		
	Join	False	×
	Data Type Database-compliant data type that mu Dialog Opening Value: timestamp	t be valid for the target database.	
		<- Update Update	e -> OK Cancel Help

8 Click on the 'Transformation' tab, double-click on Functions, double-click on Conversion and then double-click on Cast as Type (CAST) to add the Cast function to the 'Column Transformation code' in the middle pane.



9 Double-click on Available Columns, load_order_header and then order_date to enter the correct column in the 'Column Transformation Code' in the middle pane.

*	Stage Column stage_sales_detail.order	r_date ×
Properties	Target: order_date	Paste
Transformation	Source: load_order_header.order_date	Paste
	Column Transformation Code (must execute within a SQL SELECT statement).	Function Set: Default Teradata 🗸 🗸
	CAST(load_order_header.order_date AS data_type)	Image: String Image: String <t< th=""></t<>
	Word Wrap Displayed Code	
	Function Desc.:	\$
	<-	· Update Update ·> OK Cancel Help

10 Finally, replace 'data_type' with DATE FORMAT 'YYYYMMDD' and click OK.

*	Stage Column stage_sales_detail.order_date		×
Properties	Target: order_date	Paste	
Transformation	Source: load_order_header.order_date	Paste	
	Column Transformation Code (must execute within a SQL SELECT statement). CAST(load_order_header.order_date AS DATE FORMAT 'YYYYMMDI)')	Function Set Default Teradata
	Word Wrap Displayed Code		
	Function Syntax:		
	Function Desc.:		Ş
	<- Update	Update ->	OK Cancel Help

Your screen should look something like this:



In the middle pane under Source Table you can see the source of each of the columns.

You are now ready to proceed to the next step - *Including Dimension Links* (see "*1.8 Including Dimension Links*" on page 56).

1.8 INCLUDING DIMENSION LINKS

The dimension links that allow us to create the fact-like star schema now need to be included:

1 In the left pane, click on the **stage_sales_detail** table in the Stage Table object group. The middle pane should display the contents of this stage table.

stage_sales_detail Columns 😽										
Column Name	Display Name	Data Type	Source Table	Source Column						
辩 order_number	order number	decimal(12)	load_order_header	order_number						
辩 order_date	order date	date	load_order_header	order_date						
辩 customer_code	customer code	decimal(6)	load_order_header	customer_code						
辩 ship_date	ship date	date	load_order_header	ship_date						
辩 order_line_no	order line no	decimal(4)	load_order_line	order_line_no						
辩 product_code	product code	decimal(6)	load_order_line	product_code						
辩 unit_sale_price	unit sale price	decimal(9,3)	load_order_line	unit_sale_price						
辩 quantity	quantity	decimal(8)	load_order_line	quantity						
辩 sales_value	sales value	decimal(13,2)	load_order_line	sales_value						
辩 tax	tax	decimal(9,2)	load_order_line	tax						
辩 dss_update_time	dss update time	timestamp		dss_update_time						
<				>						

- **2** Drag each of the following dimensions from the right pane into the stage table in the middle pane:
 - dim_customer
 - dim_product
 - dim_order_date
 - dim_ship_date

3 This adds the dimension keys from each dimension to the stage table. Your screen should look something like this:

File Edit View Browse Ba	ickup Jobs Doc Rep	orts Validate Too	ols Window	Help				
🗄 📄 🙏 🖒 📇 🤣 Web Links 🗸	🗸 🥝 🔒 🔒 👫 🖉 Schee	luler 🧀 🕶 🧀 👻	- 🛛 🖉 🔍 🄌 li	mport 📑 👿 🥬 🤅	* 📮 🗄 🖗	•• Ŧ		
😫 Builder 💩 Scheduler 📮 Diagram 🗸 🗸 🗸								
Development • # ×	stage_sales_detail Colum	ns			4	DataWarehouse 🚽 🕂 🗙		
All Objects	Column Name	Display Name	Data Type	Source Table	Source Column	2 1 3		
Connection Onection Onection	觩 dim_customer_key	dim customer key	integer	dim_customer	dim_customer_key	- U 24		
> 🚞 Load	辩 dim_product_key	dim product key	integer	dim_product	dim_product_key	🗎 🕫 🔏 翼 🎞		
🔺 🐅 Stage	辩 dim_order_date_key	dim order date key	integer	dim_order_date	dim_order_date_key	70		
辩 stage_sales_detail	辩 dim_ship_date_key	dim ship date key	integer	dim_ship_date	dim_ship_date_key	d T deadome		
🗱 Data Store	辩 order_number	order number	decimal(12)	load_order_header	order_number	i dissuemo		
T Normalized	辩 order_date	order date	date	load_order_header	order_date	T dim_date		
a 💏 Dimension	辩 customer_code	customer code	decimal(6)	load_order_header	customer_code	T dim_order_date		
😽 dim_customer	辩 ship_date	ship date	date	load_order_header	ship_date	dim_product		
😽 dim_date	💔 order_line_no	order line no	decimal(4)	load_order_line	order_line_no	dim_ship_date		
Gim_product	💔 product_code	product code	decimal(6)	load_order_line	product_code	dss_patch		
a model_date	💔 unit_sale_price	unit sale price	decimal(9,3)	load_order_line	unit_sale_price	dss_cdirent_ddy_details		
A Fact	💔 quantity	quantity	decimal(8)	load_order_line	quantity	dss_parameter		
Join Index	💔 sales_value	sales value	decimal(13,2)	load_order_line	sales_value	<pre>dss_source_system</pre>		
J Aggregate	💔 tax	tax	decimal(9,2)	load_order_line	tax	load_customer		
a 🚚 view	💔 dss_update_time	dss update time	timestamp		dss_update_time	load_order_header		
dim_order_date						load_order_line		
Retro						model_date		
Olap Cube	<	_			>			
🥑 Olap Dimension	Results				~ म ≻	< Comparison of the second sec		
🛓 Export	Object	Message			-	·		
Procedure	dim_ship_date	COMMENT O	N COLUMN dssc	lemo.dim_ship_date.	s_holiday_flag IS			
📄 Host Script		'Flag to indica	te that the day ir	question is a holida	y. Y=holiday,			
> 🌼 Index		N=normal.' ;						
	im_ship_date	COMMENT O	N COLUMN dssc	lemo.dim_ship_date.	s_fin_quarter_no IS			
	-	Financial qua	rter number (1-4), ;				
	dim_ship_date	 COMMENT OF 'Flag to indica' 	N COLUMN dssc te if a week day \	lemo.dim_ship_date. Y=week day N=week	s_week_day_flag IS end.' ;			
	dim_ship_date	COMMENT O	N COLUMN					
	🎯 L	dssdemo.dim_	_ship_date.s_mov	/ing_fin_quarter IS 'FI	ag to indicate days			
		Set as per curr	ent fin day.'	ing back from the cu	v v	< >>		
	Results Reports					Data Wareho Source Brow		
Ready	Middle Dane: stage sale	detail Columns De	velopment (decd	amo) Userld: decde	mo Browse DataWar			

4 The stage table metadata has been defined, but the stage table has not been created. To create the stage table in the data warehouse, right-click on **stage_sales_detail** in the left pane and select **Create (ReCreate)**.



Note: The table must exist in the data warehouse before we can proceed to the next step. If the table has not been physically created then the procedure in step 5 will fail to compile.

Click **Yes** to define a Primary Index.



Type **order_number** as the Primary Index and click **OK** on the storage screen.

		Stage stage_sales_detail	
Properties	2		
Storage	4 Storage		
Override Create DDL	Database	(Default)	
N-1	Primary Index	Non-Unique Primary Index (NUPI)	N
NULES	Primary Index Name	stage_sales_detail_idx_PR	
	Primary Index Columns	order_number	
	MultiSet	True	
	Fallback	False	
	Data Block Size		•
	Enable Free Space	False	
	▲ Other		
	Optional CREATE Clause		
	Primary Index Columns		
	Primary Index Columns Columns of the Primary Index. Dialog Opening Value: sold_to_i	address_id	

7 Right-click on **stage_sales_detail**, choose **Code** and then **Build Update Procedure**.

Development 👻 म 🗙	stage_sales_detail Colum	ns				
▲ a All Objects	Column Name	Display Name	Data	Туре	Source Table	Source C
Connection	辩 dim_customer_key	dim customer key	integ	ger	dim_customer	dim_cu:
Load	辩 dim_product_key	dim product key	integ	ger	dim_product	dim_pro
🔺 辩 Stage	辩 dim_order_date_key	🗱 dim_order_date_key dim order date key integ				dim_orc
🐅 stage_sales_deta	Properties			er	dim_ship_date	dim_shi
🗱 Data Store	Storage			nal(12)	load_order_header	order_n
T Normalized	Storage			load_order_header	order_d	
Dimension	Display Columns			nal(6)	load_order_header	custom
😽 dim_customer	Display Indexes				load_order_header	ship_da
😽 dim_date	Display Data			nal(4)	load_order_line	order_li
dim_product	Quencoia Evcel			nal(6)	load_order_line	product
With the second	Query via excer			nal(9,3)	load_order_line	unit_sal
ja Pact	Report Zero Keys			nal(8)	load_order_line	quantity
	Add Column			nal(13,2)	load_order_line	sales_va
Jan Aggregate	Add Column		nal(9,2)	load_order_line	tax	
dim order date	Add Index		amp		dss_upd	
dim_bio_date	Regenerate Indexes					
■ Retro ○ Olap Cube ○ Olap Dimension Export Procedure ● Host Script ▷ Index Ready	Change Column(s) Validate Against the Da Update Comments Gather Statistics Version Control Create (ReCreate) Truncate Delete Metadata and Du Execute Update Proced Execute Custom Proced Execute Custom Proced Execute Custom Proced Documentation Projects Impact	tabase rop Table ure lure lure luer lure via Scheduler	> > >	JMN dssc row was u JMN detail.dir lorminal f ermited f JMN detail.dir	lemo.stage_sales_det updated in the data v n_order_date_key IS ' ormat is YYYYMMDE or special date handl n_customer_key IS 'G emo) UserId: dssde	ail.dss_up varehouse), but DD ing.' ; ienerated mo Brov
	Code		•	Buil	d Update Procedure	

8 Choose the **Set** based procedure generation from the stage procedure type dialog box.

Define Stage Procedure Type	×
Code can be generated for each of the following procedure types. There are advantages and disadvantages with each type, so please read the help if you are unsure of the method to choose.	Omega Cancel
 Set based insert. Assumes business key is unique. Normally the fastest method, but the least flexible. 	Set
Set based insert from all source tables (merge). A source table only needs to appear once. All source tables must have the same column names.	Set Merge
Set based insert with a Select Distinct. Do not use this option if you are including a group by statement.	Set Distinct

9 Click **OK** on the Parameters dialog.

- **10** On the Source Table Mapping screen, highlight the two tables in the left pane and click **Outer Join**.
 - Select order_number from the load_order_header empty drop-down box at the bottom of the screen.
 - Again select order_number from the load_order_line drop-down list box. This will create a join statement in the right pane.
 - Click **OK**.

	Source Table Mapping	×
Define the joins (or edit the from and wi To define a Join select two tables and	here clause). press the join type. Then select the join columns from the column lists presented.	0
Source Tables: [load_order_header] [load_order_line]	From and Where Clause: FROM [load_order_header] load_order_header LEFT OUTER JOIN [load_order_line] load_order_line ON load_order_header.order_number = load_order_line.order_number	
Outer Join Simple Join Select the columns that join the two tab [load_order_header] Word Wrap Displayed Code	ANSI join code generated les. Select the column from the Master Table first.	V

- **11** You need to match the dimension business keys with the business keys in the stage table. This associates the correct dimensional record to each stage table record. A dialog box displays for each dimensional join:
 - For the **dim customer**, select **customer code**.
 - Click > and **OK**.

	Μ	odel Busines	s Key Definition			×
			Business Keys for dim_	custome	er	
customer_code order_date order_line_no	Select the business key from the stage table that matches each business key for the model table.					
order_number product_code quantity sales_value ship_date tax unit_sale_price			Move them over to the busin correct order to match the mo	ess key li Idel busir	ist. They must be in the ness keys.	
		customer_code			code	
	>					
	<			-		
dim customer			Source Table Column List:			
		Add Text	Enter a text string and press	Add Tex	t to add a static Business Key Value	~
					OK Car	ncel

12 For **dim_product**, select **product_code**. Click **>** and **OK**.

- The business key for dim_order_date has the same column name in the stage table and the dimension view, allowing WhereScape RED to automatically move **order_date** to the left hand side.
- Click **OK** to progress to dim_ship_date, where ship_date has also been automatically chosen.
- Click **OK** again.

- **13** The final step is the population of the stage table.
 - Click on **stage_sales_detail** in the left pane, right-click and select **Execute Update Procedure**.



14 The output from the stage table being updated can now be seen in the **Results** window:

Re	sults		•	ι×
٩	Object	Message		-
0	🛨 stage_sales_detail	Table dssdemo.stage_sales_detail created.		
0	stage_sales_detail	Procedure update_stage_sales_detail rebuilt and recompiled.		
0	stage_sales_detail	Procedure Completed		
0	_ stage_sales_d	1 stage_sales_detail updated. 21 new records.		=
Re	ports Results			

You are now ready to proceed to the next step - *Creating a Fact Table* (see "*1.9 Creating a Fact Table*" on page 66).

1.9 CREATING A FACT TABLE

In this step you will create a fact table.

1 Click in the right pane and press F5 to refresh the Data Warehouse table view in the right pane.



2 Create a drop target by double-clicking on the **Fact Table object group** in the left pane.

Fact			- - - - - - - - - - -
Fact Name	Short Name	Fact Type	Database
<			>

3 Drag the stage table **stage_sales_detail** over from the right pane into the middle pane. In the following dialog, accept the name **fact_sales_detail** and click **ADD**.

	Add a New Metadata Object	×
Define the Type and	Name of the New Object.	
Specific information f	or each object type is defined in subsequent screens.	
Object Type:	Fact Table	¥
Object Name:	fact_sales_detail	
Target Location:	(local)	¥
	ADD Cance	:

4 The fact_sales_detail table Properties screen will appear. Select (**Build Procedure...**) in the update procedure drop-down and click **OK**.

×		Fact	t fact_sales_detail					×
Properties	Table Name:	fact_sales_de	etail			Table Type:	Detail	~
Storage	Unious Chart Marray							_
Override Create DDL	(maximum 22 characters)	fact_sales_de	etail					
Language Mapping	Business Display Name (EUL):	fact_sales_de	ətail					
Purpose								
Concept	Description:							^
Grain								
Examples								~
Usage	Update Procedure:	(Build Proced	lure)	~	Rebuild		Set Based Update	
Notes	Custom Procedure:	(None)		~				
	Get Key Function:	(None)		✓ Edit			Mnemonic (EUL):	
	Timostomos							
	Metadata Structure Chan	ged:	Database Created:		Data	base Altered:		
	2014-10-30 01:19:51.720	0000	2014-10-30 01:20:	14.640000	2014	-10-30 01:20:1	4.640000	
						ОК	Cancel Hel;	p

5 Select **Create and Load** to create and load the table now:

Create Data	abase Table
Fact fact_sales_detail has been del	ined
Create Create and Loa	d 🔻 Close

6 Click **Yes** to define a Primary Index on the next dialog.

•

7 Type order_number as the Primary Index and click OK.

		Fact fact_sales_detail		
Properties	<u>₽</u>			
Storage	▲ Storage			
Override Create DDL	Database	dssdemo		[
anguage Mapping	Primary Index	Non-Unique Primary Index (NUPI)		
angaage mapping	Primary Index Name	fact_sales_detail_idx_PR		
'urpose	Primary Index Columns	order_number		
Concept	MultiSet	True		
irain	Fallback	False		[
	Data Block Size			
xampies	Enable Free Space	False		[
Isage	▲ Other			
lotes	Optional CREATE Clause			
	Primary Index Columns Columns of the Primary Index. Dialog Opening Value:			
			OK Car	ncel Help

8 Select the **Business Key** for the fact table by clicking on the ellipsis button on the right hand side of the Business Key Columns field.

	fact_sale	s_detail Update Build Options.	_ □
Information	Ai		
Processing	2*		
Source	Business Key Columns		
	Parameters		
	Include Initial Load Insert	False	
	Batch Processing		
	Process by Batch	False	
	Delete Processing		
	Delete Before Insert	No	
	Update Processing		
	Process Method	Insert/Update	
	▲ Insert Method		
	Include Insert Statement	True	
	Insert New Rows Only	True	
	New Row Identification Method	Join	
	Update Method		
	Include Update Statement	True	
	Update Changed Rows Only	True	
	Change Row Identification Method	Join	
	Business Key Columns	hte processing. Dequired if include lindste options	
9 Choose order_number and order_line_no. Click > and then OK.

Columns that define the business key for update processing. Require Available Columns: dim_customer_key dim_product_key dim_order_date_key dim_ship_date_key	if include Update options. Selected Columns: order_number order_line_no
Available Columns:	Selected Columns: order_number order_line_no
dim_customer_key dim_product_key dim_order_date_key dim_ship_date_key	order_number order_line_no
order_date customer_code ship_date product_code unit_sale_price quantity sales_value tax	
< >>	< >

10 Click **OK** on the **Update Build Options** screen.

	fact_sales_	detail Update Build Options.	_ 🗆 🗙
Information	AL		
Processing			
Source	Business Key Columns	order_line_no, order_number	
	Parameters	5.1	•••
	Include Initial Load Insert	Faise	v
	Batch Processing		
	Process by Batch	False	×
	Delete Processing		
	Delete Before Insert	No	×
	Update Processing		
	Process Method	Insert/Update	~
	▲ Insert Method		
	 Include Insert Statement 	True	v
	▲ Insert New Rows Only	True	~
	New Row Identification Method	Join	*
	▲ Update Method		
	Include Update Statement	True	~
	Update Changed Rows Only	True	~
	Change Row Identification Method	Join	*
	Business Key Columns Columns that define the business key for update	e processing. Required if include Update options.	
Prev Next			OK Cancel Help

11 The output from the fact table being created and updated can now be seen in the **Results** window.

Re	sults	~ ┦ ×
٩	Object	Message
0	fact_sales_detail	Table dssdemo.fact_sales_detail created.
	fact_sales_detail	CREATE MULTISET TABLE dssdemo.fact_sales_detail, No Fallback (order_number decimal(12) TITLE 'order number', order_date date TITLE 'order date', customer_code decimal(6) TITLE 'customer code', ship_date date TITLE 'ship date', order_line_no decimal(4) TITLE 'order line no', product_code decimal(6) TITLE 'product code', unit_sale_price decimal(9,3) TITLE 'unit sale price', quantity decimal(8) TITLE 'quantity', sales_value decimal(13,2) TITLE 'sales value', tax decimal(9,2) TITLE 'tax', dim_customer_key integer TITLE 'dim customer key' NOT NULL, dim_product_key integer TITLE 'dim product key' NOT NULL, dim_order_date_key integer TITLE 'dim order date key' NOT NULL, dim_ship_date_key integer TITLE 'dim ship date key' NOT NULL, dss_update_time timestamp TITLE 'dss update time') PRIMARY INDEX fact_sales_detail_idx_PR (order_number) ;
0	 fact_sales_detail 	table comment added.
0	 fact_sales_detail 	5 column comments added.
0	 fact_sales_detail 	Procedure Completed
0	fact_sales_detail	1 fact_sales_detail updated. 21 records added. 0 records updated.
Re	ports Results	

You are now ready to proceed to the next step - *Switching to Diagrammatic View* (see "1.10 *Switching to Diagrammatic View*" on page 72).

1.10 SWITCHING TO DIAGRAMMATIC VIEW

WhereScape RED provides the ability to diagrammatically view the data warehouse you have created.

1 Click on the Diagram tab to display the **Diagram Selection** dialog.

📰 Builder 🕓 Scheduler 🛛 📑 Diagram 🗙

2 Select an object **Type** of **Fact** to narrow the selection list and then select **fact_sales_detail.** Click on the **Schema Diagram** button to display a star schema diagram.

	Diagram Selection	
Object to Diagram		
Group:	×	Schema Diagram
Project:	v	Source Diagram
Type: Fact	¥	
Object: <u>Itact_sales_detail</u>	¥	Joins Diagram
Display Columns	Link Levels: 4	Links Diagram
Restrict diagram objects t	o Group/Project	Impact Diagram
	Cancel	Dependency Diagram

The diagram will be displayed.



4 To view a more detailed diagram, click on the Toggle button You will need to use the Zoom In button to see the diagram more clearly.



TIP: To view the source tracking of the **fact_sales_detail** table, click once more on the **button**, choose the **fact_sales_detail** table and then click on the **Source Diagram** button.

Diagram Selection ×				
Object to Diagram Group:	v	Schema Diagram		
Project:	V	Source Diagram		
Type: Fact Object: fact sales detail	v 	Joins Diagram		
Display Columns	Link Levels: 4	Links Diagram		
Restrict diagram objects t	o Group/Project			
		Impact Diagram		
	Cancel	Dependency Diagram		

The diagram will be displayed.



6 To close the diagrammatic view, click on the **X** on the diagram tab, or alternatively, return to the Builder section by clicking the Builder tab.

🔡 Builder 🥴 Scheduler 🛛 🚑 Diagram

You are now ready to proceed to the next step - *Producing Documentation* (see "1.11 *Producing Documentation*" on page 77).

1.11 PRODUCING DOCUMENTATION

WhereScape RED also provides the ability to produce user and technical documentation. This is obviously of more value if the descriptive data has been entered against the columns and tables in the data warehouse, which we have not done during this tutorial.

1 To view the documentation for the components of the data warehouse, select **Doc > Create Documentation**.



2 Select a file path (directory) under which to save the HTML files that will be produced. Click **Save**.

>		Save As		×
Save in:	Documents	~	G 🤌 📂 🛄 -	
(And	Name	*	Date modified	Type 🔨
20 A	퉬 images		6/20/2014 12:21 PM	File fol
Recent places	🙆 _blank.html		6/20/2014 12:21 PM	HTML
	🧧 glossary.htm	h	6/20/2014 12:21 PM	HTML
	🗿 index.html		6/20/2014 12:21 PM	HTML
Desktop	indextech.ht	ml	6/20/2014 12:21 PM	HTML
<u> </u>	indexuser.html		6/20/2014 12:21 PM	HTML
67	🕘 maintech.html		6/20/2014 12:21 PM	HTML
Libraries	🗿 mainuser.html		6/20/2014 12:21 PM	HTML
	🕘 Names.html		6/20/2014 12:21 PM	HTML
	🕘 transforms.h	transforms.html		HTML
Computer	🖉 wsl_obj_1_te	ch.html	6/20/2014 12:21 PM	HTML
	🖉 wsl_obj_2_te	🕘 wsl_obj_2_tech.html		HTML
🛛 🦉 🦉 wsl obi 3 tech.htr		ch.html	6/20/2014 12:21 PM	HTML Y
Network	`			,
	File name:	RED Doc1.html	¥	Save
	Save as type:	Select an output directory	~	Cancel

3 The next screen allows for the inclusion of a banner and user defined links. Leave the **Links** option un-ticked and click **OK** to proceed.

Documentation Creation Options ×					
User and technical documentation will be created in HTML format in the destination directory.					
To use a custom look and feel add your own MainStyle.css file into the destination directory. Cancel					
Documentation Little (e.g. Data Warehouse):					
Do you want to link in any custom HTML pages?					
Do you want to include current table space usage? It will take longer to create the documentation.	✓ Sizes				
How would you like the columns sorted? Column Order 	O Column Name O Business Name				
Do you want to include shadows on the diagram boxes?	✓ Shadow				
Do you want to create impact analysis on load tables? It will take longer to create the documentation.	✓ Impact				
Do you want to replace the existing style sheet? Do not tick this box if you utilize a custom style sheet.	Replace Style Sheet				
Do you wish to limit the complexity of the diagrams? Select the maximum number of process steps to display in the source diagrams.	¥				

4 The documentation will run.

	Documentation Progress	×
Overall Progress:		
Step:		
Creating index pages		

TIP: To view the documentation, select **Doc > Display Documentation** on the main menu tabs

Doc	Reports	Validate	Tools	Window	
1	Create Documentation				
100	Display Documentation				
	Display Do	cumentatio	on in Bro	wser	