Q.1 Attempt any TWO question of the following : [10]

Q.1(a) What is data warehouse? List and explain the characteristics of data warehouse. [5]

(A) **Data Warehouse :** is a decision database system. It is designed to support the decision makers in the organization to analyze the historical data and to achieve the strategic level goals.

- Used for Online Analytical Processing (OLAP). This reads the historical data for the Users for business decisions.
- The Tables and joins are simple since they are de-normalized. This is done to reduce the response time for analytical queries.
- Data – Modeling techniques are used for the Data Warehouse design.
- Optimized for read operations.
- High performance for analytical queries.
- Is usually a Database.

Following are the characteristics of data warehouse

1. Subject Oriented Data
2. Integrated Data
3. Time-Referenced Data
4. Non-Volatile Data

1. **Subject Oriented :** Data warehouses are designed to help you analyze data. For example, to learn more about your company's sales data, you can build a warehouse that concentrates on sales. Using this warehouse, you can answer questions like "Who was our best customer for this item last year?" This ability to define a data warehouse by subject matter, sales in this case, makes the data warehouse subject oriented.

2. **Integrated :** Integration is closely related to subject orientation. Data warehouses must put data from disparate sources into a consistent format. They must resolve such problems as naming conflicts and inconsistencies among units of measure. When they achieve this, they are said to be integrated.

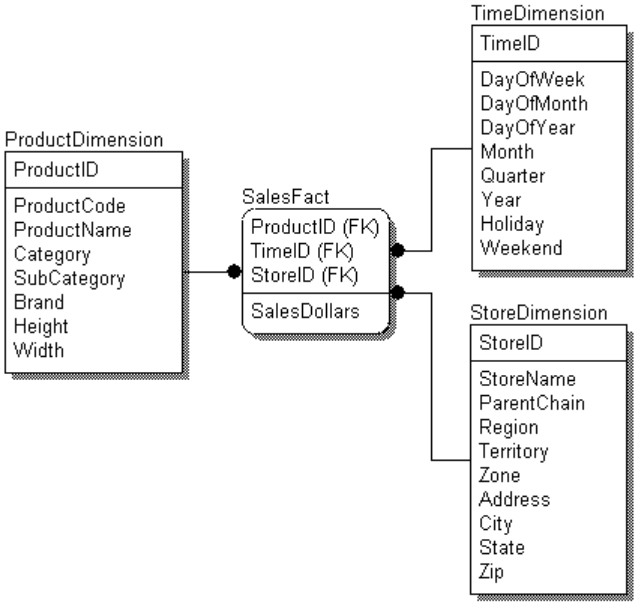
3. **Time Variant :** In order to discover trends in business, analysts need large amounts of data. This is very much in contrast to online transaction processing (OLTP) systems, where performance requirements demand that historical data be moved to an archive. A data warehouse's focus on change over time is what is meant by the term time variant.

4. **Non-volatile :** Nonvolatile means that, once entered into the warehouse, data should not change. This is logical because the purpose of a warehouse is to enable you to analyze what has occurred.

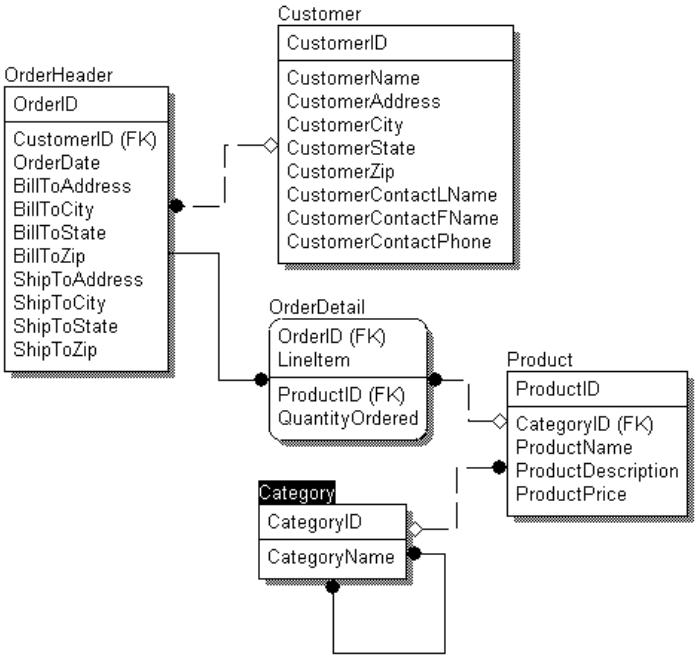
Q.1(b) Explain star schema and snow flake schema. [5]

(A) **Star Schema Model and Snow Flake Model**

- The central theme of the dimensional model is the star schema which consists of a central 'fact table', containing measures surrounded by descriptors called as 'dimensions'.
- In a star schema, a dimension is complex and contains relationships such as hierarchies; it is flattened to a single dimension.



- Another version of star scheme is a snowflake schema. In this, the complex dimensions are normalised. Here dimensions maintain relationships to other levels of the same dimensions.
- When representing the snowflake schema, 'category' and 'brand' are kept as separate entities but are related to 'product'.



Q.1(c) Differentiate between operational system and informational system. [5]

(A) The following table summarizes main differences between OLPT and OLAP:

	OLTP	**OLAP**
Application	Operational: ERP, CRM, legacy apps, …	Management Information System, Decision Support System
Typical users	Staff	Managers, Executives
Horizon	Weeks, Months	Years
Refresh	Immediate	Periodic
Data model	Entity-relationship	Multi-dimensional
Schema	Normalized	Star
Emphasis	Update	Retrievel

Q.1(d) Why a dimension is called Slowly changing dimension? Explain. [5]

(A) Slowly changing Dimension refers to the fact that dimension values will change over time. Although this doesn't happen often, they will change and hence the 'slowly' designation. We have to specify how we want to handle the change we will have the following three choices, which are related to the issue of whether or how we want to maintain a history of that change in the dimension.

- **Type 1**: Do not keep a history. This means we basically do not care what the old value was and just change it.

- **Type 2**: Store the complete change history. This means we definitely care about keeping that change along with any change that has ever taken place in the dimension.

- **Type 3**: Store only the previous value. This means we only care about seeing what the previous value might have been, but don't care what it was before that.

Q.2Attempt any TWO question of the following : [10]

Q.2(a) What are different steps used for configuring a listener in Oracle Warehouse Builder. [5]

(A) Configuring the listener

The listener is the utility that runs constantly in the background on the database server, listening for client connection requests to the database and handling them. It can be installed either before or after the creation of a database, but there is one option during the database creation that requires the listener to be configured–so we'll configure it now, before we create the database.

Run Net Configuration Assistant to configure the listener. It is available under the Oracle menu on the Windows Start menu as shown in the following image:
The welcome screen will offer us four tasks that we can perform with this assistant. We'll select the first one to configure the listener.

The next screen will ask you what we want to do with the listener. The four options are as follows:

- Add
- Reconfigure
- Delete
- Rename

Only the Add option will be available since we are installing Oracle for the first time. The remainder of the options will be grayed out and will be unavailable for selection. If they are not, then there is a listener already configured and we can proceed to the next section–Creating the database.

The next screen is the protocol selection screen. It will have TCP already selected for us, which is what most installations will require. This is the standard communications protocol in use on the Internet and in most local networks. Leave that selected and proceed to the next screen to select the port number to use. The default port number is 1521, which is standard for communicating with Oracle database and is the one most familiar to anyone who has ever worked with an Oracle database. So, change it only if you want to annoy the Oracle people in your organization who have all memorized the default Oracle port of 1521.

That is the last step. It will ask us if we want to configure another listener. Since we only need one, we'll answer "no" and finish out the screens by clicking on the Finish button back on the main screen.

Q.2(b) Explain Oracle Warehouse Builder architecture with diagram. [5]

(A) OWB components and architecture

Now that we've installed the database and OWB client, let's talk about the various components that have just been installed that are a part of the OWB and the architecture of OWB in the database. Then we'll perform one final task that is required before using OWB to create our data warehouse.

Oracle Warehosue Builder is composed on the client of the Design Center (including the Control Center Manager) and the Repository Browser. The server components are the Control Center Service, the Repository (including Workspaces), and the Target Schema. A diagram illustrating the various components and their interactions follows:

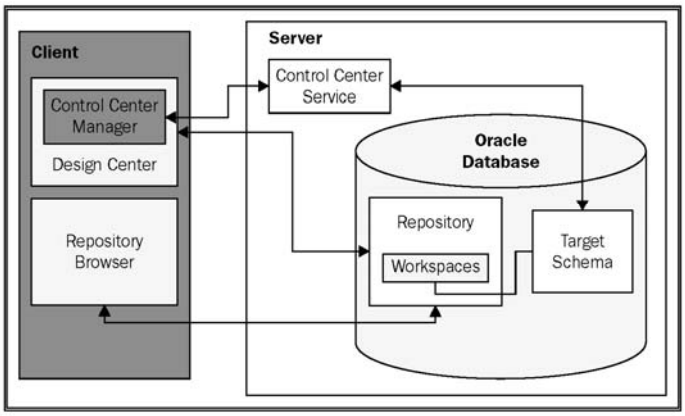


Fig. 1

The Design Center is the main client graphical interface for designing our data warehouse. This is where we will spend a good deal of time to define our sources and targets, and describe the extract, transform, and load (ETL) processes we use to load the target from the sources. The ETL procedures are what we will define to carry out the extraction of the data from our sources, any transformations needed on it and subsequent loading into the data warehouse. What we will create in the Design Center is a logical design only, not a physical implementation. This logical design will be stored behind the scenes in a Workspace in the Repository on the server. The user interacts with the Design Center, which stores all its work in a Repository Workspace.

We will use the Control Center Manager for managing the creation of that physical implementation by deploying the designs we've created into the Target Schema. The process of deployment is OWB's method for creating physical objects from the logical definitions created using the Design Center. We'll then use the Control Center Manager to execute the design by running the code associated with the ETL that we've designed. The Control Center Manager interacts behind the scenes with the Control Center Service, which runs on the server as shown in the previous image. The user directly interacts with the Control Center Manager and the Design Center only.

The Target Schema is where OWB will deploy the objects to, and where the execution of the ETL processes that load our data warehouse will take place. It is the actual data warehouse schema that gets built. It contains the objects that were designed in the Design Center, as well as the ETL code to load those objects. The Target Schema is not an actual Warehouse Builder software component, but is a part of the Oracle Database. However, it will contain Warehouse Builder components such as synonyms that will allow the ETL mappings to access objects in the Repository.

The Repository is the schema that hosts the design metadata definitions we create for our sources, targets, and ETL processes. Metadata is basically data about data. We will be defining sources, targets, and ETL processes using the Design Center and the information about what we have defined (the metadata) is stored in the repository.

Q.2(c) Explain three components of design center. [5]

(A) Three Components of design center

Before discussing the project in more detail, let's talk about the three windows in the main Design Center screen. They are as follows:

- Project Explorer
- Connection Explorer
- Global Explorer

The Project Explorer window is where we will work on the objects that we are going to design for our data warehouse. It has nodes for each of the design objects we'll be able to create. It is not necessary to make use of every one of them for every data warehouse we design, but the number of options available shows the flexibility of the tool. The objects we need will depend on what we have to work with in our particular situation. In our analysis earlier, we determined that we have to retrieve data from a database where it is stored.

The Connection Explorer is where the connections are defined to our various objects in the Project Explorer. The workspace has to know how to connect to the various databases, files, and applications we may have defined in our Project Explorer. As we begin creating modules in the Project Explorer, it will ask for connection information and this information will be stored and be accessible from the Connection Explorer window. Connection information can also be created explicitly from within the Connection Explorer.

There are some objects that are common to all projects in a workspace. The Global Explorer is used to manage these objects. It includes objects such as Public Transformations or Public Data Rules. A transformation is a function, procedure, or package defined in the database in Oracle's procedural SQL language called PL/SQL. Data rules are rules that can be implemented to enforce certain formats in our data.

Q.2(d) Write the steps to import metadata from a file. [5]

(A) Step 1 : First we need to perform or create a module that will contain our file definition. Navigate to

My Project → Database → File node

right click and select New to launch the wizard.

Step 2 : When we click on the Next button on the welcome screen, the Great module wizard only ask for a name and description. Will enter name of ACME files and click on next button.

Step 3 : We need to edit the connection in step 2 we will click on edit button, a new wizard will appear and it will ask is for a name, description and the path to the folder where the files are.

Step 4 : The name field is prefilled. with the suggested name based on the module name, it only adds that number 1 to the end i.e.
ACME_FILES_LOCATION

Step 5 : Notice the Type drop down menu. It has 2 entries. General and FTP if we select FTP (file Transfer Protocol) it will ask more information i.e. Host name, username, password and path the username and password is the name of the computer.

Step 6 : The simplest option us to store the file on the same computer on which we are running the database and click on ok button.

Step 7 : We will then check the box for Import after finish and click on the finish button.

The import metadata wizard appears next and the steps are as follows :
Step 1 : The first screen is for selecting filter information
There are 2 options.

- All Data files
- Data files matching this pattern
  We will choose all data files and click on next button.

Step 2 : The next screen is the object selection screen , which is similar to the one we used for selecting the database objects to import from we click on the plus sign beside the folder name in that our file counties .csv file.

Step 3 : We select the counties .scv file click on the right arrow (>) and then mouse on to the next step.

Step 4 : It provides the summary and Impact page of the database imports.

- One difference is the red X in the status column.
- The red X indicates that we have not yet specified that information for a file.

Step 5 : We will use sample button to enter that information. When we click on that button, we will have flat file sample wizard on welcome screen, we will click on next button to move on to step 1. The following one the options on this screen.

- Help button for more information
- Character set is language related.
- Number of character to sample determine how much of the file the wizard will read to get an idea.

Step 6 : In the next step, we will specify even more information about the characteristic of a file.
Here we will specify record information. The commas determine where one column ends and other begins. CR (carriage.return) will indicate the end of a row.

Step 7 : The next step is where we specify the file format. Which tells the wizard how the columns are separated. Our records are separated by comma.

Step 8 : In this we get to specify the number of rows to skip and whether the file contains more than one record type two options are available.

- single Record
- Multi Record

The record are all single Record so we will select first option.

Step 9 : This is where we specify the details about what each field contains and give each field a name.

Step 10 : Here we will get a summary screen of what the wizard will do and click on finish button.

Step 11 : Click on the finish button on the step 3 screen of the import. Metadata wizard. It will create our file module index the file node.

Step 12 : We will make sure to select the save. All from the Design menu in the Design centre to save the metadata.

Q.3Attempt any TWO question of the following : [10]

Q.3(a) Explain multidimensional implementation of data warehouse. [5]

(A) Multidimensional Implementation

- A multidimensional implementation or OLAP (online analytic or analytical processing) requires a database with special features that allow it to store cubes as actual objects in the database, and not just tables that are used to represent a cube and dimensions.
- It also provides advanced calculation and analytic content built into the database to facilitate advanced analytic querying.
- These kinds of analytic databases are well suited to providing the end user with increased capability to perform highly optimized analytical queries of information. Therefore, they are quite frequently utilized to build a highly specialized data mart, or a subset of the data warehouse, for a particular user community.

Q.3(b) Explain how to create target user and target module. [5]

(A) Creating a target user and module

Create a target user

There are a couple of ways we can go about creating our target user—create the user directly in the database and then add to OWB, or use OWB to physically create the user. If we have to create a new user, and if it's on the same database as our repository and workspaces, it's a good idea to use OWB to create the user, especially if we are not that familiar with the SQL command to create a user. However, if our target schema were to be in another database on another server, we would have to create the user there. It's a simple matter of adding that user to OWB as a target, which we'll see in a moment. Let's begin in the Design Center under the Global Explorer. There we said it was for various objects that pertained to the workspace as a whole.

Right-click on the Users node and select New.. to launch the Create User dialog box.

With this dialog box, we are creating a workspace user. We create a workspace user by selecting a database user that already exists or create a new one in the database. If we already had a target user created in the database, this is where we would select it. We're going to click on the Create DB User... button to create a new database user.

We need to enter the system username and password as we need a user with DBA privileges in the database to be able to create a database user. We then enter a username and password for our new user. As we like to keep things basic, we'll call our new user ACME_DWH, for the ACME data warehouse. We can also specify the default and temporary table space for our new user, which we'll leave at the defaults. The dialog will appear like the following when completely filled in:

The new user will be created when you click on the OK button, and will appear in the righthand window of the Create User dialog already selected for us. Click on the OK button and the user will be registered with the workspace, and we'll see the new username if we expand the User node under Security in the Global Explorer. We can continue with creating our target module now that we have a user defined in the database to map to.

Q.3(c) Explain how to create a time dimension using time dimension wizard. [5]

(A) Creating a Time dimension with the Time Dimension Wizard

1. The first step of the wizard will ask us for a name for our Time dimension. We're going to call it DATE_DIM. If we try to use just DATE, it will give us an error message because that is a reserved word in the Oracle Database; so it won't let us use it.
2. The next step will ask us what type of storage to use for our new dimension.
3. Now this brings us to step 3, which asks us to specify the data generation information for our dimension. The Time Dimension Wizard will be automatically creating a mapping

for us to populate our Time dimension and will use this information to load data into it. It asks us what year we want to start with, and then how many total years to include starting with that year. The numbers entered here will be determined by what range of dates we expect to load the data for, which will depend on how much historical data we will have available to us. We have checked with the DBAs for ACME Toys and Gizmos Company to get an idea of how many years' worth of data they have and have found out that there is data going back three years. Based on this information, we're going to set the start year to 2007 with the number of years set to three to bring us up to 2009.

4. This step is where we choose the hierarchy and levels for our Time dimension. We have to select one of the two hierarchies. We can use the Normal Hierarchy of day, month, quarter, and year; or we can choose the Week Hierarchy, which consists of two levels only—the day and the calendar week. Notice that if we choose the Week Hierarchy, we won't be able to view data by month, quarter, or year as these levels are not available to us. This is seen in the following image:
5. Let's move on to step 5 where the wizard will provide us the details about what it is going to create. An example is shown in the following image, which is what you should see if you've made all the same selections as we've moved along.
6. Continuing to the last step, it will display a progress bar as it performs each step and will display text in the main window indicating the step being performed. When it completes, we click on the Next button and it takes us to the final screen—the summary screen. This screen is a display of the objects it created and is similar to the previous display in step 5 of 6 that shows the pre-create settings. At this point, these objects have been created and we press the Finish button. Now we have a fully functional Time dimension for our data warehouse.

Q.3(d) Explain different steps of creation of a cube in Oracle Warehouse Builder. [5]

(A) Creating a cube in OWB

Creating a cube with the wizard

We will start the wizard in a similar manner to how we started up the Dimension wizard. Right-click on the Cubes node under the ACME_DWH module in Project Explorer, select New, and then Using Wizard... to launch the cube-creation wizard. The first screen will be the welcome screen, which will summarize the steps.

The following are the steps in the creation process:

1. We proceed right to the first step where we give our cube a name. As we will be primarily storing sales data, let's call our cube SALES and proceed to the next step.
2. In this step, we will select the storage type just as we did for the dimensions. We will select ROLAP for relational storage to match our dimension storage option, and then move to the next step.
3. In this step, we will choose the dimensions to include with our cube. We have defined three, and want all them all included. So, we can click on the double arrow in the center to move all the dimensions and select them. If we had more dimensions defined than we were going to include with this cube, we would click on each, and click on the single right arrow (to move each of them over); or we could select multiple dimensions at one time by holding down the Ctrl key as we clicked on each dimension. Then click the single right arrow to move those selected dimensions.
4. Moving on to the last step, we will enter the measures we would like the cube to contain. When we enter QUANTITY for the first measure and SALES_AMOUNT for the second one, we end up with a screen that should look similar to this with the dialog box expanded to show all the columns:

Q.4Attempt any TWO question of the following : [10]

Q.4(a) Write a short note on ETL process. [5]

(A) ETL

The process of extracting, transforming, and loading data can appear rather complicated. We do have a special term to describe it, ETL, which contains the three steps mentioned. We're dealing with source data on different database systems from our target and a database from a vendor other than Oracle. Let's look from a high level at what is involved in getting that data from a source system to our target, and then take a look at whether to stage the data or not. We will then see how to automate that process in Warehouse Builder, which will relieve us of much of the work.

Manual ETL processes

First of all, we need to be able to get data out of that source system and move it over to the target system. We can't begin to do anything until that is accomplished, but what means can we use to do so? We know that the Oracle Database provides various methods to load data into it. There is an application that Oracle provides called SQL*Loader, which is a utility to load data from flat files. This could be one way to get data from our source system. Every database vendor provides some means of extracting data from their tables and saving it to flat files. We could copy the file over and then use the SQL*Loader utility to load the file. Reading the documentation that describes how to use that utility, we see that we have to define a control file to describe the loading process and definitions of the fields to be loaded. This seems like a lot of work, so let's see what other options we might have.

However, our target database structure doesn't look anything like the source database structure. The POS Transactional database is a relational database that is highly normalized, and our target consists of cubes and dimensions implemented relationally in the database. How are we going to get the data copied into that structure? Clearly, there will be some manipulation of the data to get it reformatted and restructured from source to target. We cannot just take all the rows from one table in the source structure and copy them into a table in the target structure for each source table. The data will have to be manipulated when it is copied. This means we need to develop code that can perform this rather complex task, depending on the manipulations that need to be done.

In a nutshell, this is the process of extract, transform, and load. We have to:

1. Extract the data from the source system by some method.
2. Load flat files using SQL*Loader or via a direct database link. Then we have to transform that data with SQL or PL/SQL code in the database to match and fit it into the target structure.
3. Finally, we have to load it into the target structure.

The good news here is that the Warehouse Builder provides us the means to design this process graphically, and then generate all the code we need automatically so that we don't have to build all that code manually.

Q.4(b) Explain different components of mapping editor in Oracle Warehouse Builder. [5]

(A) Mappings and operators in OWB

- Mapping: The Mapping window is the main working area on the right where we will design the mapping. This window is also referred to as the canvas. The Data Object Editor used the canvas to lay out the data objects and connect them, whereas this editor lays out the mapping objects and connects them. This is the graphical display that will show the operators being used and the connections between the operators that indicate the data flow from source to target.

- Explorer: This window is similar to the Project Explorer window from the Design Center, and is the same as the window in the Data Object Editor. It has the same two tabs—the

Available Objects tab and the Selected Objects tab. It displays the same information that appears in the Project Explorer, and allows us to drag and drop objects directly into our mapping. The tabs in this window are:

- Available Objects: This tab is almost like the Project Explorer. It displays objects defined in our project elsewhere, and they can be dragged and dropped into this mapping.
- Selected Objects: This tab displays all the objects currently defined in our mapping. When an object is selected in the canvas, the Selected Objects window will scroll to that object and highlight it. Likewise, if we select an object in the Selected Objects tab, the main canvas will scroll so that the object is visible and we will select it in the canvas. Go ahead and click on a few objects to see what the tab does.

- Mapping properties: The mapping properties window displays the various properties that can be set for objects in our mapping. It was called the Configuration window in Data Object Editor. When an object is selected in the canvas to the right, its properties will display in this window.

- Palette: The Palette contains each of the objects that can be used in our mapping. We can click on the object we want to place in the mapping and drag it onto the canvas. This list will be customized based on the type of editor we're using. The Mapping Editor will display mapping objects. The Data Object Editor will display data objects.

- Bird's Eye View: This window displays a miniature version of the entire canvas and allows us to scroll around the canvas without using the scroll bars. We can click and drag the blue-colored box around this window to view various portions of the main canvas. The main canvas will scroll as we move the blue box. Go ahead and give that a try. We will find that in most mappings, we'll quickly outgrow the available space to display everything at once and will have to scroll around to see everything. This can come in very handy for rapidly scrolling the window.

Q.4(c) Write the steps to connect source table to target using joiner operator with example. [5]

(A) Connecting operators is a matter of clicking and dragging a line from an output attribute of one operator to an input attribute of another operator or from one output group in one operator to an input group in another operator.

- If we connected two attributes groups together we are telling the mapping. Editor to go ahead and connect every attribute in the group.
- If we have several attributes this is a convenient way to connect them.
- So click and drag. INOUTGRP 1 of the ITEMS table operator onto the ITEMS group of the JOINER.
- Immediately it will add all the attributes from the ITEMS table to the ITEMS group in the JOINER and connect each one with a line.
- Alternatively we could have click and dragged a line from each attribute in the ITEMS table and dropped it on the ITEMS group in the JOINER.
- But it was quicker and more efficient to drag the entire group even through we won't be using every attribute.
- Let's repeat the same procedure with POS-TRANSACTIONS, REGISTERS STORES and REGIONS tables.
- Now we will drag INOUT GRPI groups to the corresponding. group is the JOINER for each table to connect them.

Q.4(d) Explain the following operators: **[5]**

(i) Cube operator **(iii) External Table operator** **(v) Constant operator**

(ii) Dimension operator **(iv) Table operator**

(A) (i) Cube operator

An operator that represent a cube in our mapping in known as cube operator.

(ii) Dimension operator

- An operator that represent the dimensions is called dimension operator.
- We have created three dimension i.e. time, product, store so to look at the example of dimension operator in DATE_DIM_MAP.

(iii) External Table operator

- This operator represents external tables.
- They can be used to access data stored in flat files as if they were tables.

(iv) Table operator

- This operator represent a table in the database.
- We will need to store data in tables in our oracle database at some point in the loading of data.

(v) Constant operator :

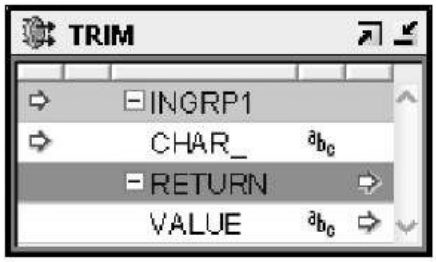
- Represents a constant value that is needed.
- It can be used to load a default value for a field that does't have any input from another source for instance.

Q.5Attempt any TWO question of the following : **[10]**

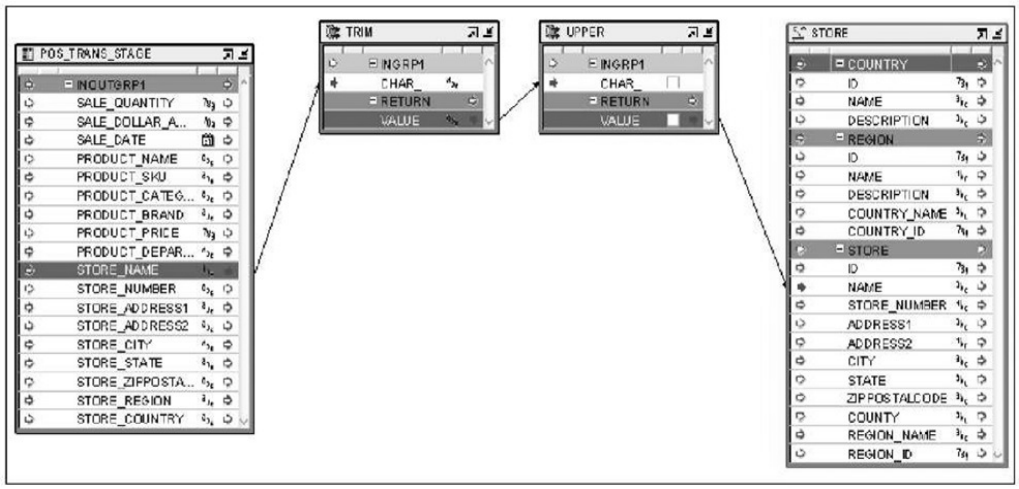
Q.5(a) What is the role of TRIM(), UPPER(), SUBSTR() operator and TO_NUMBER() in ETL mapping? **[5]**

(A) TRIM( ) – this function falls into the Transformation Operator on the mapping.

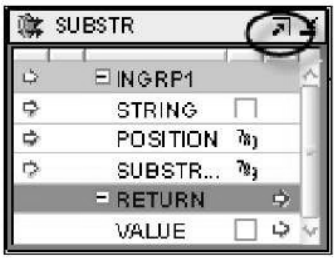
TRIM function displays the TRIM Transformation operator window on the mapping. A TRIM operator has one input attribute and one output attribute. The input attribute is the string to be trimmed for the whitespaces and the output attribute represents the result of applying the TRIM operator to the input string. It looks like the following screenshot:



UPPER( ) - this function also falls into the Transformation Operator on the mapping which converts the input string into Upper case.



SUBSTR( ) - The Transformation operators in OWB include a substr (or substring) transformation that will extract the specified number of characters from the source string. The substr transformation takes three parameters — the string we want to extract the substring from, a number indicating the start position of the substring within the string, and a number indicating the length of the substring to extract.



TO_NUMBER( ) – This function converts the expression into number.
This operator needs three parameters, only one of which is absolutely necessary — the expression we wish to convert to a number. The other two parameters are optional and include a format string that we can use if we have a particular format of number we want (such as a decimal point in a certain place) and a parameter that allows us to set a certain national language format to default to if it's different from the language set in the database.

Q.5(b) Explain product mapping with diagram. [5]

(A) **Store mapping** : Let's begin by creating a new mapping called STORE_MAP. We'll follow the procedure in the previous chapter to create a new mapping. In the Design Center, we will right-click on the Mappings node of the ACME_DW_PROJECT | Databases | Oracle | ACME_DWH database and select New.... Enter STORE_MAP for the name of the mapping and we will be presented with a blank Mapping Editor window. In this window, we will begin designing our mapping to load data into the STORE dimension.

Adding source and target operators : In the last chapter, we loaded data into the POS_TRANS_STAGE staging table with the intent to use that data to load our dimensions and cube. We'll now use this POS_TRANS_STAGE table as our source table. Let's drag this table onto the mapping from the Explorer window. Review the Adding source tables section of the previous chapter for a refresher if needed.

The target for this mapping is going to be the STORE dimension, so we'll drag this dimension from Databases | Oracle | ACME_DWH | Dimensions onto the mapping and drop it to the right of the POS_TRANS_STAGE table operator. Remember that we build our mappings from the left to the right, with source on the left and target on the right. We'll be sure to leave some space between the two because we'll be filling that in with some more operators as we proceed.

Now that we have our source and target included, let's take a moment to consider the data elements we're going to need for our target and where to get them from the source.
We might be tempted to include the ID fields in the above list of data elements for populating, but these are the attributes that will be filled in automatically by the Warehouse Builder. The Warehouse Builder fills them using the sequence that was automatically created for us when we built the dimension. We don't have to be concerned with connecting any source data to them.

There is another attribute in our STORE dimension that we haven't accounted for yet—the COUNTY attribute. We don't have an input attribute to provide direct information about it. It is a special case that we will handle after we take care of these more straightforward attributes and will involve the lookup table that we discussed earlier in the introduction of this chapter.

We're not going to directly connect the attributes mentioned in the list by just dragging a line between each of them. There are some issues with the source data that we are going to have to account for in our mapping. Connecting the attributes directly like that would mean the data would be loaded into the dimension as is, but we have investigated the source data and discovered that much of the source data contains trailing blanks due to the way the transactional system stores it. Some of the fields should be made all uppercase for consistency.

Q.5(c) Explain validation in Oracle Warehouse Builder. [5]

(A) **Validating :** We briefly touched upon the topic of validation of mapping when we were working on our SALES cube mapping and talked about the error we would get if we tried to map a date attribute to a number attribute. Error checking is what validation is for. The process of validation is all about making sure the objects and mappings we've defined in the Warehouse Builder have no obvious errors in design.

Let's recap how we go about performing a validation on an object we've created in the Warehouse Builder. There are a number of places we can perform a validation. One of them is the main Design Center.

Validating in the Design Center : There is a context menu associated with everything we create. You can access it on any object in the Design Center by right-clicking on the object of your choice. Let's take a look at this by launching our Design Center, connecting as our ACMEOWB user, and then expanding our ACME_DW_PROJECT. Let's find our staging table, POS_TRANS_STAGE, and use it to illustrate the validation of an object from the Design Center. As we can recall, this table is under the ACME_DWH module in Oracle node and right-clicking on it will present.

The validation will result in one of the following three possibilities:

1. The validation completes successfully with no warnings and/or errors as this one did.
2. The validation completes successfully, but with some non-fatal warnings.
3. The validation fails due to one or more errors having been found.

In any of these cases, we can look at the full message by double-clicking on the Message column in the window on the right. This will launch a small dialog box that contains the full text of the message.

Validating from the editors : The Data Object Editor and the Mapping Editor both provide for validation from within the editor. We'll talk about the Data Object Editor first and then about the Mapping Editor because there are some subtle differences.

Validating in the Data Object Editor : Let's double-click on the POS_TRANS_STAGE table name in the Design Center to launch the Data Object Editor so that we can discuss validation in the editor. We can right-click on the object displayed on the Canvas and select Validate from the pop-up menu, or we can select Validate from the Object menu on the main editor menu bar. Another option is available if we want to validate every object currently loaded into our Data Object Editor. It is to select Validate All from the Diagram menu entry on the main editor menu bar. We can also press the validate icon on the General toolbar, which is circled in the following image of the toolbar icons:

Fig.

Validating in the Mapping Editor

We'll now load our STORE_MAP mapping into the Mapping Editor and take a look at the validation options in that editor. Unlike the Data Object Editor, validation in the Mapping Editor involves all the objects appearing on the canvas together because that is what composes the mapping. We can only view one mapping at a time in the editor. The Data Object Editor lets us view a number of separate objects at the same time.

For this reason, we won't find a menu entry that says "Validate All" in the Mapping Editor. What we will find in the main editor menu bar's Mapping menu is an entry called Validate that we can select to validate this mapping. To the right of the Validate menu entry we can see the F4 key listed, which means we can validate our mapping by pressing the F4 key also. The key press we can do to validate is not available in the Data Object Editor because it wouldn't know which object we intend to validate. We also have a toolbar in the Mapping Editor that contains an icon for validation, like the one we have in the Data Object Editor. It is the same icon as shown in the previous image of the Data Object Editor toolbar.

Let's press the F4 key or select Validate from the Mapping menu or press the Validate icon in the toolbar. The same behavior will occur as in the Data Object Editor—the validation results will appear in the Generation Results window.

Q.5(d) Write a short note on Control Center Manager. [5]

(A) The Control Center Manager window overview

The Object Details window

Let's click on the ACME_DWH_LOCATION again in the left window and look at the complete list of objects for our project. The statuses will vary depending on whether we've done any deployments or not, when we did them, and whether there are any warnings or failures due to errors that occurred. If you're following along exactly with the book, the only deployment we've done so far is the POS_TRANS_STAGE table and the previous image of the complete Control Center manager interface shows it as the only one that has been deployed successfully. The remainder all have a deploy status of Not Deployed.

The columns displayed in the Object Details window are as follows:

- Object: The name of the object
- Design Status: The status of the design of the object in relation to whether it has been deployed yet or not
  - New: The object has been created in the Design Center, but has not been deployed yet
  - Unchanged: The Object has been created in the Design Center and deployed previously, and has not been changed since its last deployment
  - Changed: The Object has been created and deployed, and has subsequently undergone changes in the Design Center since its last deployment subsequently undergone changes in the Design Center since its last deployment.
- Deploy Action: What action will be taken upon the next deployment of this object in the Control Center Manager
  - Create: Create the object; if an object with the same name already exists, this can generate an error upon deployment
  - Upgrade: Upgrade the object in place, preserving data
  - Drop: Delete the object
  - Replace: Delete and recreate the object; this option does not preserve data
- Deployed: Date and time of the last deployment
- Deploy Status: Results of the last deployment
  - Not Deployed: The object has not been deployed yet
  - Success: The last deployment was successful, without any errors or warnings
  - Warning: The last deployment had warnings
  - Failed: The last deployment failed due to errors

- Location: The location defined for the object, which is where it will be deployed
- Module: The module where the object is defined.
  Some of the previous columns will allow us to perform an action associated with the column by double-clicking or single-clicking in the column. The following is a list of the columns that have actions available, and how to access them:
- Object: Double-click on the object name to launch the appropriate editor on the object.
- Deploy Action: Click on the deploy action to change the deploy action for the next deployment of the object via a drop-down menu. The list of available actions that can be taken will be displayed. Not all the previously listed actions are available for every object. For instance, upgrade is not available for some objects and will not be an option for a mapping.

The Control Center Jobs window

Every time we do a deployment or execute a mapping, a job is created by the Control Center to perform the action. The job is run in the background while we can continue working on other things, and the status of the job is displayed in the Control Center Jobs window. Looking back at the previous image of the Control Center manager, we can see the status of the POS_TRANS_STAGE table deployment that we performed. The green check mark indicates it was successful. If we want to see more details, especially if there were warnings or errors, we can double-click on the line in the Control Center Jobs window and it will pop up a dialog box displaying the details. An example of the dialog box is shown next:

Q.6Attempt any TWO question of the following : [10]

Q.6(a) Write a short note on Snapshot feature of Oracle Warehouse Builder. [5]

(A) Snapshots

A snapshot captures all the metadata information about an object at the time the snapshot is taken and stores it for later retrieval. It is a way to save a version of an object should we need to go back to a previous version or compare a current version with a previous one. We take a snapshot of an object from the Design Center by right-clicking on the object and selecting the Snapshot menu entry. This will give us three options to choose from as shown next:

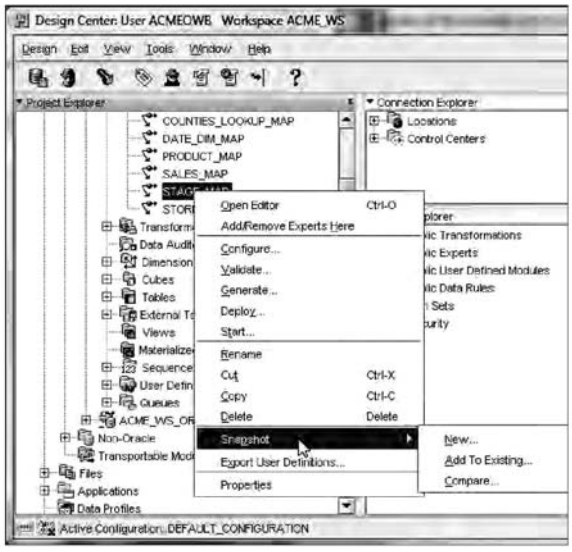


Fig. 1

That same menu entry is available on the main menu of the Design Center under the Design entry. We can create a new snapshot, add this object to an existing snapshot, or compare this object with an already saved snapshot. The last option is particularly useful for seeing what has changed since the snapshot was taken.

Let's first take a snapshot of our POS_TRANS_STAGE table in the new project we created in the last section. We'll right-click on the table name and select Snapshot | New...

to create a new snapshot of it. This will launch the Snapshot Wizard to guide us through the three-step process as shown next:

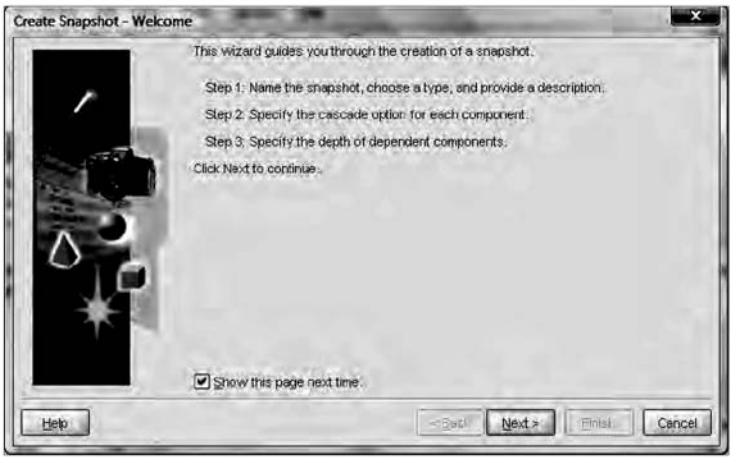


Fig. 2

1. We'll click on the Next button to move to step 1 of the wizard where we'll give our snapshot a name. This name will be stored in the database as an object name, and so must conform to the Oracle Database requirement that identifiers be no more than 30 characters in length and also must be unique. The wizard will validate the name for us and pop up an error message if we've violated any of the naming conventions, including exceeding the 30 character limit or using invalid characters such as spaces. We'll call our snapshot POS_TRANS_STAGE_SNAP. If we like, we can enter a description also to further identify what is in the snapshot.

   There are two types of snapshots we can take: a full snapshot that captures all metadata and can be restored completely (suitable for making backups of objects) and a signature snapshot that only captures the signature or characteristics of an object just enough to be able to detect changes in an object. The reason for taking the snapshot will generally dictate which snapshot is more appropriate. We can click on the Help button on this screen to get a detailed description of the two options. In our case, we'll take a full snapshot this time. Full snapshots can be converted to signature snapshots later if needed, and can also be exported like regular workspace objects. Having selected Full, we click on the Next button to move to the next step.

2. This step displays a list of the objects we're capturing in this snapshot. We have the option on this screen to select Cascade, which applies to folder-type objects. We can take a snapshot of any workspace object, including nodes and even the entire project itself. We can then select Cascade to have it include every object contained within that folder object. This is an easy way to capture a large number of objects at once. In our case, we're only capturing the POS_TRANS_STAGE table, so Cascade would have no effect. We'll click on Next and move to step 3, the final step.

3. In the final step we are asked to select a depth to which we'd like to traverse to capture any dependent objects for this object. The screenshot of this step is shown next:

Fig. 3

- **Restore:** We can restore a snapshot from here, which will copy the snapshot objects back to their place in the project, overwriting any changes that might have been made. It is a way to get back to a previously known good version of an object if, for example, some future change should break it for whatever reason.
- **Delete:** If we do not need a snapshot anymore, we can delete it. However, be careful as there is no recycle bin for deleted snapshots. Once it's deleted, it's gone forever. It will ask us if we are sure before actually deleting it.
- **Convert to Signature:** This option will convert a full snapshot to a signature snapshot.
- **Export:** We can export full snapshots like we can export regular workspace objects. This will save the metadata in a file on disk for backup or for importing later.
- **Compare:** This option will let us compare two snapshots to each other to see what the differences are.

Q.6(b) Write a short note on Synchronization in Oracle Warehouse Builder. [5]

(A) Synchronizing : Many operators we use in a mapping represent a corresponding workspace object. If the workspace object (for instance, a table) changes, then the operator also needs to change to be kept in sync. The process of synchronization is what accomplishes that, and it has to be invoked by us when changes are made.

Now that we have the updated table definition for the POS_TRANS_STAGE table, we have to turn our attention to any mappings that have included a table operator for the changed table because they will have to be synchronized to pick up the change. These operators are bound to an actual table using a table definition like we just edited. When the underlying table definition gets updated, we have to synchronize those changes with any mappings that include that table. We now have our STAGE_MAP mapping copied over to our new project. So let's open that in the mapping editor by double-clicking on it and investigate the process of synchronizing.

When we open the mapping for the first time, it may have all the objects overlapping in the middle of the mapping. So we'll just click on the auto-layout button, If we look at the POS_TRANS_STAGE mapping operator, we can scroll down the INOUTGRP1 attribute group or maximize the operator to view all the attributes to see that the new STORE_AREA_SIZE column is not included.

To update the operator in the mapping to include the new column name, we must perform the task of synchronization, which reconciles the two and makes any changes to the operator to reflect the underlying table definition. We could just manually edit the properties for the operator to add the new column name, but that still wouldn't actually synchronize it with the actual table. Doing the synchronization will accomplish both—add the new column name and synchronize with the table. In this particular case there is another reason to synchronize, and that is we copied this mapping into the new project from

another mapping where it was already synchronized with tables in that project. This synchronization information is not automatically updated when the mapping is copied.

To synchronize, we right-click on the header of the table operator in the mapping and select Synchronize... from the pop-up menu, or click on the table operator header and select Synchronize... from the main menu Edit entry, or press the F7 key. This will pop up the Synchronize dialog box as shown next:

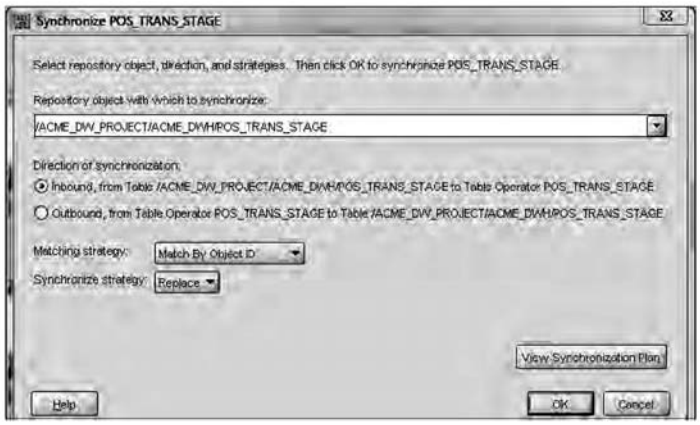


Fig. 1

Now we can see why it's so important to make sure we're in the correct project. From the entry indicating the repository object from which it will synchronize, we can see that it's still set to point to the original POS_TRANS_STAGE table in the ACME_DW_ PROJECT project and not the new one we just edited in this project. If we were to rely upon this, we would think we are in the wrong project. We need to click on the dropdown menu and select the POS_TRANS_STAGE table in our new COPY_MODULE. In fact, this new copy module is the only one we have available. This is good because we wouldn't want to select an object in another module. It's only set that way in this case because it was just copied from that other project. However, we can tell something is a little strange there because the path listed for the POS_TRANS_STAGE table stops at ACME_DW_PROJECT and no icon is displayed for the type of object. When we select the POS_TRANS_STAGE table in our new project, we get the correct display as shown next:

Fig. 2

This looks much better. Notice how the path includes the workspace name now to fully place it in context. It knows what kind of object it is, a table, so it can display the correct icon. Now we can proceed with deciding whether this will be inbound or outbound.

Changes to dimensional objects and auto-binding

When we created our dimensional objects (dimensional in this context being used to refer to both dimensions and cubes), we saw how it automatically created a matching relational table for us. This would be used to hold the data for the object because we had selected ROLAP for the storage implementation. This table was then bound to the dimension with dimension attributes, and levels bound to columns in the table.

Q.6(c) Differentiate between single hypercube v/s multicube. [5]

(A) Single Hypercube vs. Multicube

Just as in a relational database, where the data is typically stored in a series of two dimensional tables, the data in an OLAP database consists of values with differing dimensionality. For example, consider the case of a bank which wants to analyze profitability by customer and product. Certain revenues may be dimensioned by customer, product, time, geography and scenario. But costs such as back office salaries might be dimensioned by cost type, cost center and time. Thus, a real world multidimensional database must be capable of storing data that has differing dimensionality in a single logical database that presents a single conceptual view to the user.

It is not acceptable to insist that all data in the model be dimensioned by the same criteria. Putting data into an element called "no product" because the data item happens not to be dimensioned by product, merely serves to confuse the user. Products that cannot support a multicube architecture with the ability to logically relate cubes, force unreasonable compromises on the developer of sophisticated models. Not only does it make the access to the data less logical but it can have serious adverse effects on database size and retrieval efficiency.

Q.6(d) Write a short note on data sparsity. [5]

(A) Data Sparsity And Data Explosion

The reality of data explosion in multidimensional databases is a surprising and widely misunderstood phenomenon. For those about to implement a project with considerable exposure to OLAP products, it is critically important to understand what data sparsity and data explosion are, what causes these, and how these can be avoided for, the consequences of ignoring data explosion can be very costly and, in most cases, result in project failure.
Input data or base data (i.e. before calculated hierarchies or levels) in OLAP applications is typically sparse (not densely populated). Also, as the number of dimensions increase, data will typically become sparser (less dense).

For example, in a one-dimensional matrix, you can suppress all zero values and, therefore, have a 100 percent dense matrix. In a two dimensional matrix, you cannot suppress zeros if there is a non-zero value in any element in the two dimensions (see figure 1 and 2).

While it is not true in all cases, typically, as the number of dimensions in a model increases, so does the data sparsity. For example, if you are storing sales data by product and by month, it is conceivable that you will sell each product each month (100 percent dense).

	Year
Product A	10
Product B	20
Product C	8
Product D	15

Fig. 1: 100 Per cent Dense Structure

However, if you were storing sales data, by product, by customer, by region, by month, clearly you would not sell each product to every customer in every region every month.

	Q1	Q2	Q3	Q4
Product A	10	0	0	0
Product B	0	20	0	0
Product C	0	0	8	0
Product D	0	0	0	15

Fig. 2 : 25 Per cent Dense Structure–4 out of 16 data Points Populated

By adding the dimension "gender" to this model, you would double the possible size of the cube by storing the data by either of the two variables male or female, but the size of the stored data would remain the same. In this case, by introducing another simple dimension, the sparsity will have doubled!

Data explosion is the phenomenon that occurs in multidimensional models where the derived or calculated values significantly exceed the base values. There are three main factors that contribute to data explosion.

1. Sparsely populated base data increases the likelihood of data explosion.
2. Many dimensions in a model increase the likelihood of data explosion, and
3. A high number of calculated levels in each dimension increase the likelihood of data explosion.

Let's extend our previous example to explain this. In figure 3, the 100 percent dense one-dimensional model with two levels has base data that exceeds calculated data in a ratio of 4:1. These is no data explosion here.

In Figure 4, the 25 percent dense two-dimensional model with three levels in one dimension has base data of four values that "explodes" to 15 calculated values.

By increasing sparsity and/or adding dimensions and/or adding calculated levels in dimensions in the example shown in figure 37, the data explosion rate will increase.

	Year
Product A	10
Product B	20
Product C	8
Product D	15
Total	53

Fig. 3 : 100 Per cent Dense Structure–No Data Explosion

	Year	Q1	Q2	Q3	Q4
Product A	**10**	10	0	0	0
Product B	**20**	0	20	0	0
Product C	**8**	0	0	8	0
Product D	**15**	0	0	0	15
Product A + Product B	**30**	10	20	0	0
Product C + Product D	**23**	0	0	8	15
Product A + Product B + Product C + Product D	**53**	10	20	8	15

Fig. 4 : Data Explosion is 3.75 Times

Q.7Attempt any THREE question of the following : [15]

Q.7(a) Discuss the different types of facts with respect to measures stored in the fact table in a Data Warehouse. [5]

(A) **(i) Additive** : A fact data said to be fully additive if it is additive over every dimension of its dimensionality

(ii) Non-Additive : Measures that are not additive over any dimension

(iii) Semi-Additive : Measures that can be added across some dimensions

The values that quantify facts are usually numeric, and are often refused to as measures. Measures are typically additive along all dimensions such as quantity in a sales fact table.

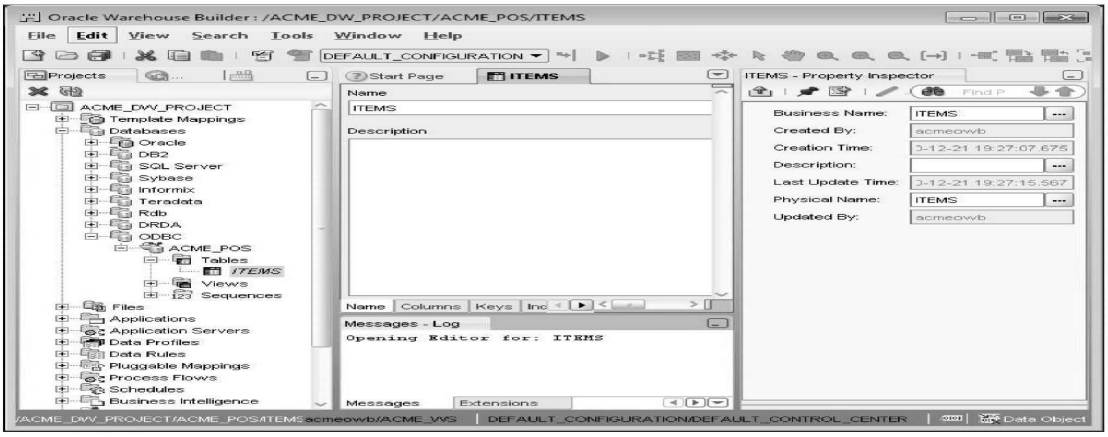
In the real world, it is possible to have a fact table that contains no measures or facts. These tables are called 'factless fact tables' or 'junction tables'

Q.7(b) Explain the procedure for defining source metadata manually with Data Object Editor. [5]

(A) 1. To start building our source tables for the POS transactional SQL Server database, Launch the OWB Design Center. Expand the ACME_DW_PROJECT node. Search the already created ACME_POS module for the SQL Server source database under the Databases | ODBC node so that is where we'll create the tables. Navigate to the Databases | ODBC node, and then select the ACME_POS module under this node. We will create our source tables under the Tables node, so let's right-click on this node and select New Table from the pop-up menu. As no wizard is available for creating a table, we are using the Data Object Editor to do this.

2. The first screen we'll be presented with is a small popup asking us to fill in the name and a description for the new table we're creating. We're going to create the metadata for the ITEMS table so let's change the name to ITEMS and click OK to continue.

3. Upon selecting OK, we are presented with the Table Editor screen on the right hand side of the main Design Center interface. It's a clean slate that we get to fill in, and will look similar to the following screenshot:



Q.7(c) Explain the tabs Name, storage, Attributes, Levels and Hierarchies in Editor Window of any object that is currently being edited. [5]

(A) **Name :** Display the name of the dimensions.

Storage : What storage option is set for our dimension object in the database, whether relational or multidimensional

Attributes: It displays the attributes as a tabular form allowing to view or edit them, including adding new attributes or deleting the existing ones.

Levels: Here are can view and /or edit the levels for our dimension. We can check and uncheck boxes to indicate which of the various level types we want to use and which attributes are applicable to which level.

Hierarchies : If let us specify hierarchy information for our dimensions and will even let us create a new hierarchy. It is possible that we may have selected more levels on the previous page and now need to assign them to a hierarchy.

Q.7(d) What is staging in data warehouse? List the advantages and disadvantages of staging. [5]

(A)
- Staging is the process of copying the source data temporarily into a tables in our target database.
- Here we can perform any transformation that are required before loading the source data into the final target tables.

To stage or not to stage :

The points to consider to keep the process flowing as fast as possible are amount of source data we will be dealing with

- amount of manipulations of the source data.
- If the source data is in another database other than an oracle database.

Advantages :

- If it is appropriate, you can take a snapshot of your production database and then do you ETL from the restored backup or snapshot. This could save load on your production database.
- You may need complicated processing for your ETL which requires many intermediate tables which have no use except for the ETL process. You may not want to clutter your DW with these intermediary tables.
- Your raw data may not be available all at once and our need some where to accumulate it before starting your ETL process to build your data warehouse.
- Your data warehouse may have production window requirement which can't be met by your ETL and so your need to stage your "Output" rather than or in addition to your production database.

Q.7(e) What is the role of a LOOKUP operator in a mapping? [5]

(A) Lookup operator supports multiple lookup object – more efficient use of screen real estate.
Lookup operator supports complex lookup Conditions including a range of conditions types such as = , < , > or BETWEEN - AND (any boolean expression basically)
Preferences for which row to return in case of multiple rows in lookup result (ANY, FIRST, LAST, nTH).
The steps to create a lookup table are

(i) Right click on the Mappings nodes, select New…, enter COUNTIES lookup_MAP in the name field, and click on the ok button.

(ii) In the Mapping Editor that pops up, let's drag an External Table Operator from the Palette window onto the mapping.

(iii) On the Add External Table operator pop–up window that appears, our COUNTIES external table is visible. We will select that and click on the ok button to continue. This will drop an External Table Operator on our mapping that is bound to our external table.

(iv) We need to get that data loaded into a regular table in the database, so next we will drag a Table Operator onto the mapping.

(v) In the resulting Add Table operator pop up that appears, we specify what table we want to add.

(vi) We will click on ok and it will drop a Table operator onto our mapping with no attributes defined in it.

Q.7(f) Differentiate between ROLAP and MOLAP. [5]

(A)

	ROLAP	MOLAP
1.	It is used when DW contains relational data.	It is used when DW. DW contains relational as well as non-relational data.
2.	In this different cubes will created dynamically.	In this different cubs will got already stored.
3.	It does not contains prefabricated cubes.	It contains different prefabricated cubes.
4.	It consists of analytical server.	It contains MDDB server.
5.	It is comparatively slower than MOLAP.	It is comparatively faster than ROCAP.
6.	It consumes less amount of memory.	It consumes more amount of memory.
7.	The architecture is simple and easy to implement.	The architecture is different to implement.

❑ ❑ ❑ ❑ ❑