In this reading sample, we’ll use Chapter 4 to show you how the book is structured to prepare you for the exam. Chapter 4 provides details on the core modeling concepts you will find throughout the book and the exam. Understanding the basics, such as views, joins, cubes, fact tables, hierarchies, CDS views, and the different information views, is critical in developing more advanced data modeling skills.

“Information Modeling Concepts”

Contents

Index

The Authors

Rudi de Louw

SAP HANA Certification Guide

559 Pages, 2016, $79.95

www.sap-press.com/3859
**Information Modeling Concepts**

**Techniques You’ll Master**
- Understand general data modeling concepts like views, cubes, fact tables, and hierarchies
- Know when to use the different types of joins in SAP HANA
- Examine the differences between attributes and measures
- Learn about the different types of information views that SAP HANA uses
- Get to know projections, aggregations, and unions used in SAP HANA information views
- Describe Core Data Services (CDS) in SAP HANA
- Get a glimpse of best practices and general guidelines for data modeling in SAP HANA
Before moving onto the more advanced modeling concepts, it’s important to understand the basics. In this chapter, we’ll discuss the general concepts of information modeling in SAP HANA. We’ll start off by reviewing views, join types, cubes, fact tables, hierarchies, the differences between attributes and measures, the different types of information views that SAP HANA offers, and CDS views. From there, we’ll take a deeper dive into SAP HANA’s information views, as well as concepts like calculated columns, how to perform currency conversions, input parameters, decision tables, and so on.

Real-World Scenario

You start a new project. Many of the project team members have some knowledge of traditional data modeling concepts and describe what they need in those terms. You need to know how to take those terms and ideas, translate them into SAP HANA modeling concepts, and implement them in SAP HANA in an optimal way.

Some of the new concepts in SAP HANA are quite different than what you’re used to—for example, not storing the values of a cube, instead calculating it when required. If you understand these modeling concepts, you can quickly understand and create real-time applications and reports in SAP HANA.

Objectives of This Portion of the Test

The objective of this portion of the SAP HANA certification is to test your understanding of basic SAP HANA modeling artifacts.

For the certification exam, SAP HANA modelers must have a good understanding of the following topics:

- The different types of joins available in SAP HANA and when to use each one
- How the SAP HANA information views correspond to traditional modeling artifacts
- The different types of SAP HANA views and when to use them
- The purpose of Core Data Services (CDS) in SAP HANA

Key Concepts Refresher

This section looks at some of the core data modeling concepts used when working with SAP HANA and that will be covered on the exam. Let’s start from where the data is stored in SAP HANA by looking at tables.

Tables

Tables allow you to store information in rows and columns. Inside SAP HANA, there are different ways of storing data. We can either store it in a table as row-oriented or column-oriented. In a normal disk-based database, we use row-oriented storage because it works faster with transactions in which we are reading and updating single records. However, because SAP HANA works in memory, we prefer the column-oriented method of storing data in the tables, with which we can make use of compression in memory, remove the storage overhead of indexes, save a lot of space, and optimize for performance by loading compressed data into computer processors.

Once we have our tables, we can begin combining them in SAP HANA information models.

Views

The first step in building information models is building views. A view is a database entity that is not persistent and is defined as the projection of other entities, like tables.

For a view, we take two or more tables, link these different tables together on matching columns, and select certain output fields. Figure 4.1 illustrates this process.
There are four steps when creating database views:

1. **Select the tables**
   Select two or more tables that have related information—for example, two tables listing the groceries you bought. One table contains the shop where you bought each item, the date, the total amount, and how you paid for your groceries. The other table contains the various grocery items you bought.

2. **Link the tables**
   Next, link the selected tables on matching columns. In our example, perhaps our two tables both have shopping spree numbers, should ideally a unique number for every shopping trip you took. We call this a **key field**. We can link the tables together with **joins** or **unions**.

3. **Select the output fields**
   We want to show a certain number of columns that are of interest to us—for example, to use in our grocery analytics. Normally, you don’t want all the available columns as part of the output.

4. **Save the view**
   Finally, save your view, and it’s created in the database. These views are sometimes called **SQL views**. We can even add some filters on our data in the view—for example, to show only the shopping trips in 2015 or all the shopping trips in which we bought apples.

When we call this view, the database gathers the data. The database view pulls the tables out, links them together, chooses the selected output fields, reads the data in the combined fields, and returns the result set. After this, the view “disappears” until we call it again. Then, the database deletes all of the output from the view. It does not store this data from the view inside the database storage.

**Tip**

It’s important to realize that database views don’t store the result data. Each time you call a view, it performs the full process again.

You might have also heard about **materialized views**, in which people store the output created by a view in another table. However, in our definition of views, we stated that a view is a database entity that is **not persistent**; that is the way we use the term **view** in SAP HANA.

The data stays in the database tables. When we call the view, the database gathers the subset of data, showing us only the data that we asked for. If we ask a database view for that same data a few minutes later, the database will regather and recalculate the data again.

This concept is quite important going forward, because you make extensive use of views in SAP HANA. SAP HANA creates a cube-like view, for example, sends the results back to us, and “disappears” again. SAP HANA performs this process fast, and we avoid consuming a lot of extra memory by not storing static results. What really makes this concept important is the way it enables us to perform **real-time reporting**.
In the few minutes between running the same view twice, our data in the table might have changed. If we use the same output from the view every time, we will not get the benefit of seeing the effect of the updated data. Extensive caching of result sets does not help when we want real-time reporting.

Note
Real-time reporting requires recalculating results, because the data can keep changing. Views are designed to handle this requirement elegantly.

We have discussed the basic type of database views here, but SAP HANA takes the concept of views to an entirely new level! Before we get there, we’ll look at a few more concepts that we will need for our information modeling journey.

Cardinality
When we join tables together, we need to define how much data we expect to get from the various joined tables. This part of the relationship is called the cardinality.

There are four basic cardinalities that you will typically work with. The cardinality is expressed in terms of how many records on the left side join to how many records on the right side.

- **One-to-one**
  Each record on the left side matches with one record on the right side. For example, say that you have a lookup table of country codes and country names. In your left table, you have a country code called ZA. In the right table (the lookup table), this matches with exactly one record, showing that the country name of South Africa is associated with ZA.

- **One-to-many**
  In this case, each record on the left side matches with many records on the right side. For example, say you have a list of publishers in the left table. For one publisher—such as SAP PRESS—we find many published books.

- **Many-to-one**
  This is the inverse of one-to-many. For example, this case may apply when we have many people in a small village all buying items at the local corner shop.

- **Many-to-many**
  For example, this case may apply when many people read many web pages.

Joins
Before we look at the different types of joins, let’s quickly discuss the idea of “left” and “right” tables. Sometimes, it does not make much of difference which table is on the left of the join and which table is on the right, because some join types are symmetrical. However, with a few join types it does make a difference.

We recommend putting the most important table, seen as the main table, on the left of the join. An easy way to remember which table should be the table on the right side of the join is to determine which table can be used for a lookup or for a dropdown list in an application (see Figure 4.2).

![Figure 4.2 Table Used for Dropdown List on Right-Hand Side of a Join](image)
This does not always mean that this table has to be physically positioned on the left of the screen in our graphical modeling tools. When we create a join in SAP HANA, the table we start the join from (by dragging and dropping) becomes the “left table” of the join.

Databases are highly optimized to work with joins; they are much better at it than you can hope to be in your application server or reporting tool. SAP HANA takes this to the next level with its new in-memory and parallelization techniques.

In SAP HANA, there are a number of different types of joins. Some of these are the same as what you would find in traditional databases, but others are unique to SAP HANA. Let’s start by looking at the basic join types.

**Basic Join Types**

The first basic join types that you will find in most databases are inner joins, left outer joins, right outer joins, and full outer joins. We will discuss SQL in more detail in Chapter 8.

The easiest way to visualize these join types is to use circles, as shown in Figure 4.3. This is a simplified illustration of what these join types do. (The assumption is that the tables illustrated here contain unique rows; that is, we join the tables via primary and foreign keys, as illustrated in Figure 4.1. If the tables contain duplicate records, this visualization does not hold.)

The left circle represents the table on the left side of the join. In Figure 4.3, the left table contains the two values A and B. The right circle represents the table on the right side of the join and contains the values B and C.

**Inner Join**

The inner join is the most widely used join type in most databases. When in doubt and working with a non-SAP HANA database, try an inner join first.

The inner join returns the results from the area where the two circles overlap. In effect, this means that a value is shown only if it is found to be present in both the left and the right tables. In our case, in Figure 4.3, you can see that the value B is found in both tables.

Because we are only working with the “overlap” area, it does not matter for this join type which table is on the left or on the right side of the join.

**Left Outer Join**

When using an inner join you may find that you’re not getting all the data back that you require. To retrieve the data that was left out, you would use a left outer join. You will only use this join type when this need arises.

With a left outer join, everything in the table on the left side is shown (first table). If there are matching values in the right table, these are shown. If there are any “missing” values in the right hand table, these are shown with a NULL value.

For this join type, it is important to know which tables are on the left and the right side of the join.
Right Outer Join
The inverse of the left outer join is the right outer join. This shows everything from the right table (second table). If there are matching values in the left table, these are shown. If there are any “missing” values in the left hand table, these are shown with a NULL value.

Full Outer Join
As of SAP HANA SPS 11, we now have the full outer join. Many other databases also have this join type, so it is still regarded as one of the four basic join types. This join type combines the results sets of both the left outer join and right outer join into a single result set.

Self-Joins
There isn’t really a join type called a self-join; the term refers to special cases in which a table is joined to itself. The same table is both the left table and the right table in the join relationship. The actual join type would still be something like an inner join.

This configuration is used mostly in recursive tables—that is, tables that refer back to themselves, such as in HR organizational structures. All employee team members have a manager, but managers are also employees of their companies and have someone else as their manager. In this case, team members and managers are all employees of the same company and are stored in the same table. Other examples include cost and profit center hierarchies or bills of materials (BOMs).

We return to this concept when we look at the Hierarchies section later in this chapter.

SAP HANA Join Types
The join types on the right side of Figure 4.4 are unique to SAP HANA: referential join, text join, and temporal join.

Referential Join
The referential join type normally returns exactly the same results as an inner join. So, what’s the difference? There are many areas SAP HANA tries to optimize the performance of queries and result sets. In this case, even though a referential join gives the same results as an inner join, it provides a performance optimization under certain circumstances.

The referential join uses referential integrity between the tables in the join. Referential integrity is used in a business system to ensure that data in the left and right tables match. For example, we can’t have an invoice that isn’t linked to a customer, and the customer must be inserted into the database before we are allowed to create an invoice for that customer. Note that you do not need a value
on both sides of the join: You may have a customer without an invoice, but you may not have an invoice without a customer.

If we can prove that the data in our tables has referential integrity, which most data in financial business systems has, then we can use a referential join. In this case, if we’re not reading any data from the right table, then SAP HANA can quite happily ignore the entire right table and the join itself, and it doesn’t have to do any checking, which speeds up the whole process.

**Tip**
A referential join is the optimal join type in SAP HANA.

### Text Join
Another SAP HANA-specific joint type is a text join. The name gives a clue as to when we will use this type of join.

The right-hand table, as shown in Figure 4.5, would be something like a text lookup table—for example, a list of country names. On one side, we would have a country code such as US or DE, and in the text lookup table we would have the same country code and would link it with the key, and also would have the actual name of the country (e.g., United States or Germany).

SAP sells software in many countries, and SAP business systems are available in multiple languages, and the name of the country and be translated into different languages. The fourth column in the text lookup table, called the language code, indicates into which language the country name has been translated. For example, for the country called DE and a language code of EN, the name of the country is in English and thus would be Germany. If the language code was DE (for German), the name of the country would be Deutschland, and if the language code was ES (for Español, indicating Spanish) then the name of the country would be Alemania.

Therefore, the same country can have totally different names depending on what language you speak.

In such a case, SAP HANA does something clever: By looking at the browser, the application server, or the machine that you are working on, it determines the language that you are logged on in. If you are logged in using English, it knows to use the name Germany. If you are logged on using German, it provides the German country name Deutschland for you.

#### Figure 4.5  Text Join

<table>
<thead>
<tr>
<th>Field 1</th>
<th>Country code</th>
<th>Field n</th>
<th>Language code</th>
<th>Country name</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>DE</td>
<td>Germany</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>DE</td>
<td>Deutschland</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>DE</td>
<td>Alemania</td>
</tr>
<tr>
<td></td>
<td>US</td>
<td></td>
<td>EN</td>
<td>United States</td>
</tr>
<tr>
<td></td>
<td>ZA</td>
<td></td>
<td>EN</td>
<td>South Africa</td>
</tr>
</tbody>
</table>

The text join behaves like a left outer join but always has a cardinality of one-to-one (1:1). In other words, it only returns a single country name based on the language you’re logged on with. Even if you’re logged in via mobile phone, your mobile phone has a certain language setting.

### Temporal Join
We use the temporal join when we’ve got FROM and TO date and time fields, or integers.

For example, say that you’re storing the fuel price for gasoline in a specific table. From the beginning of this month to the end of this month, gasoline sells for a certain price. Next month, the price differs, and maybe two weeks later, it’s adjusted again. Later, you’ll have a list of all the different gasoline prices over time.

As a car owner, you now want to perform calculations for how much you’ve paid for your gasoline each time you filled up your tank. However, you just have the number of gallons and the dates of when you filled up.
You can say, “OK, on this date, I filled up the tank.” You then have to look up which date range your filling-up date falls within and find the price for that specific date. You can then calculate what you paid to fill your tank for each of your dates. Figure 4.6 illustrates this example.

This date range lookup can be a little more complicated in a database. Sometimes, programmers read the data into an application server and then loop through the different records. In this case, a temporal join makes it easy, because SAP HANA will perform the date lookups in the FROM and the TO fields automatically for you, compare it to the date you’ve supplied, and get you the right fuel price at that specific date. It will perform all the calculations automatically for you—a great time and effort saver.

A temporal join uses either a referential join; or an inner join to do the work. A temporal join requires valid to and from dates (in the gasoline price table) and a valid date-time column (in your car log book table).

**Key Concepts Refresher**

**Spatial Join**
Spatial joins became available in SAP HANA SPS 09. SAP HANA provides spatial data types, which we can use, for example, with maps. These all have the prefix ST_. A location on a map, with longitude and latitude, would be an ST_POINT data type. (We will discuss spatial data and analytics further in Chapter 9.)

We can use special spatial joins between tables in SAP HANA (see Figure 4.7). Say that you have a map location stored in a ST_POINT data type in one table. In the other table, you have a list of suburbs described in a ST_POLYGON data type. You can now join these two tables with a spatial join and define the spatial join to use the ST_CONTAINS method. This will calculate for each of the locations (ST_POINT) in which suburb (ST_POLYGON) they are located (ST_CONTAINS).

There are about a dozen methods like ST_CONTAINS available for spatial joins. For example, you can test if a road crosses a river, if one area overlaps another (e.g., mobile phone tower reception areas), or how close certain objects are to each other.
We have only mentioned the two-dimensional aspects in relation to maps, but many of these spatial functions can be used in three dimensions.

**Dynamic Joins**

A dynamic join is not really a join type; it’s a join property. It is also a performance enhancement available for SAP HANA. Once you have defined a join, you can mark it as a dynamic join. For this to work, you have to join the tables on multiple columns.

Let’s assume you define a static (normal) join on columns A, B, and C between two tables, as shown in Figure 4.8. In this case, the join criteria on all three columns will be evaluated every time the view is called in a query.

If you now mark the join as *dynamic*, the join criteria will only be evaluated on the columns involved in the query to the view. If you only ask for column A in your query of the view, then only column A’s join criteria will be evaluated. The join criteria on columns B and C will not be evaluated, and your system performance improves.

**Tip**

If your query does not request any of the join columns in the dynamic join, it will produce a runtime error.

**Core Data Services Views**

CDS is a modeling concept that is becoming more important with every new release of SAP HANA. To understand why we need CDS, it is important to remember the wider context of how SAP HANA systems are deployed.

While learning SAP HANA, you work with a single system, but productive environments normally include development, quality assurance, and production systems. You perform development in the development environment, in which you have special modeling and development privileges. Your models and code then enter the quality assurance system for testing. When everyone is satisfied with the results, the models and code move to the production system. You do not have any modeling and development privileges in the quality assurance and production systems; you need a way to create views and other database objects in the production system, ideally without giving people special privileges.

An SAP HANA system can be deployed as the database for an SAP business system. In such a case, ABAP developers do not have any modeling and development privileges, even in the development system. We need a way to allow ABAP developers to create views and other database objects in the production system without these special privileges.

With that in mind, let’s look at why CDS is a good idea with the following example:

You need to create a new schema in your SAP HANA database. It’s easy to use a short SQL statement to do this, but remember the wider context. Someone will need the same create privileges in the production system! We want to avoid this, because giving anyone too much authorization in a production system can weaken security and encourage hackers.

CDS allows for a better way to create such a schema: Specify the schema creation statement once by creating a CDS (text) file with the `schema_name="DEMO";`
instruction. When we take this CDS file to the production system, the SAP HANA system automatically creates the schema for us. In the background, SAP HANA automatically converts the CDS file into the equivalent schema creation SQL statement and executes this statement.

The system administrators do not get privileges to create any schemas in the production system. They have rights to import CDS files, but they have no control over the contents of the CDS files; that’s decided by the business. CDS thus gives us a nice way to separate what the business needs from what database administration can do.

There’s much more to CDS. You can specify table structures (metadata) and associations (relationships) between tables with CDS, and taking this CDS file to production creates a new table. Even more impressive, if you want to modify the table structure later, SAP HANA does not delete the existing tables and recreate them; it automatically generates and executes the correct SQL statements to modify only the minimal changes to the existing table structures—for example, adding only a new column. CDS does not replace SQL statements, because it ultimately translates back into SQL statements. CDS adds the awareness to use a table creation SQL statement the first time it’s run but a table modification SQL statement the second time it’s run.

We can also create views using CDS. These CDS views can be used as data sources in our SAP HANA information views. Please note that these CDS views do not replace the more powerful information views we can create in SAP HANA.

ABAP developers can also use CDS inside ABAP. When a CDS view is sent to the SAP HANA system, it creates the necessary SAP HANA database objects without the ABAP developer having to log into the SAP HANA database. Because ABAP is database-independent, the same CDS file sent to another database will create the database objects relevant to that database. Because of this database independence, the versions of CDS for ABAP and for SAP HANA have slight differences.

The example screens in this book make use of the SHINE demo package described in Chapter 2. SHINE is a good example of CDS in action. You do not have to create a schema, create tables, or import data into these tables. When SHINE is imported into your SAP HANA system, it automatically creates all that content for you by using CDS statements.

Now, let’s take our knowledge of these concepts to the next “dimension.”

Cube
One of the most important modeling concepts we will cover is the cube. We’ll expand on this concept when we discuss the information views available in SAP HANA. In this section, we’ll use a very simplistic approach to describing cubes to give you an understanding of the important concepts.

When you look at a cube the first time, it looks like multiple tables that are joined together—and at a database level, that might be true. What makes a cube a cube are the “rules” for joining these tables together.

The tables you join together contain two types of data: transactional data and master data. As shown in Figure 4.9, we have a main table in the middle called a fact table. This is where our transactional data is stored. The transactional data stored in the fact table is referred to as either key figures or measures. In SAP HANA, we just use the term measures.

![Figure 4.9 Cube Features](image-url)
characteristics or attributes, or sometimes even facets. In SAP HANA, we just use the term attributes.

To clear up these statements, let’s walk through an example. Our cube example will be simplified to communicate the most basic principles.

In our example, we’ll work with the sentence “Laura buys 10 apples.” You will agree that this sentence describes a financial transaction: Laura went to the shop to buy some apples, she paid for them, and the shop owner made some profit.

Where will we store this transaction’s data in the cube? There are a couple of “rules” that we have to follow. The simplest rule is that you store the data with which you can perform calculations on in the fact table, and you store everything else that you can’t perform calculations on in the dimension tables.

In our transaction of “Laura buys 10 apples,” we first look at Laura. Can we perform a calculation on Laura? No, we can’t. Therefore, we should put her name into one of the dimension tables. We put her name in the top dimension table in Figure 4.10. Laura is not the only customer of that shop; her name is stored with those of many other people in this dimension table. We will call this table the Customer dimension table.

Next, let’s look at the apples. Can we perform calculations on apples? No, we can’t, so again this data goes into a dimension table. The shop doesn’t sell only apples; it also sells a lot of other products. The second dimension table is therefore called the Product dimension table.

In an enterprise data warehouse, we would normally limit the number of dimension tables that we link to the fact table for performance reasons. Typically, in an SAP BW system, we limit the number of dimension tables to a maximum of 16 or 18.

Finally, let’s look at the number of apples, 10. Can we perform a calculation on that? Yes, because it’s a number. We therefore store that number in the fact table.

To complete the picture, if Laura buys more apples, then she is a top customer; let’s say she’s customer number 2. Apples are something the store keeps in stock, so apples are product number 7.

If we store a link (join) to customer 2 and another link (join) to product 7 and the number 10 (number of apples), we would say that that constitutes a complete transaction. We can abbreviate the transaction “Laura buys ten apples” to 2, 10, and 7 in our specific cube.

Laura and apples are part of our master data. The transaction itself is stored in the fact table. We’ve now put a real-life transaction into the cube.

Hopefully, this gives you a better idea of what a cube is, and how to use it. Let’s complete this short tour of cubes by clarifying a few terms ahead.

Attributes and Measures

As previously shown in Figure 4.9, attributes are stored in dimension tables, and measures are stored in fact tables. Let’s define attributes and measures:

- **Attributes**
  Attributes describe a transaction, like the customer Laura or the product apples. This is also referred to as master data. We cannot perform calculations on attributes.

- **Measures**
  Measures are something we can measure and perform calculations on. Measurable (transactional) information goes into the fact table.
Fact Tables in the Data Foundation

Figure 4.10 showed only a single fact table, but sometimes we have more complex cubes that include multiple fact tables. In SAP HANA, we call a group of fact tables (or the single fact table) the data foundation.

When we join the data foundation (fact tables) to the dimension tables, we always put the data foundation on the left side of the join.

Star Joins

The join between the data foundation and the dimension tables is normally a referential join in SAP HANA. We refer to this as a star join, which indicates that this is not just a normal referential join between low-level tables, but a join between a data foundation and dimension tables. This is seen as a join at a higher level.

Originally, we called these logical joins in SAP HANA, however they were renamed star joins as of SAP HANA SPS 10.

With all the building blocks in place, let’s start examining SAP HANA information views.

Information Views

Let’s start our introduction to SAP HANA information views by looking back at what we learned when we discussed cubes.

Dimension Views

The first thing we need to build cubes is the master data. Master data is stored in dimension tables. Sometimes, these dimension tables are created for the cubes. In SAP HANA, we do not create dimension tables as separate tables; instead we build them as views.

We join all the low-level database tables that contain the specific master data we are interested in into a view. For example, we might join two tables in a view to build a new Products dimension in SAP HANA, and maybe another three tables to build a new Customers dimension.

In this book, we will refer to these types of views simply as dimension views. In SAP HANA, there are currently two types of dimension views available:

- **Attribute views**
  The older version of dimension views are called attribute views. This name refers to the data, called attributes, stored in dimension tables. Attribute views are still used in exceptional cases in SAP HANA SPS 11.

- **Calculation views of type dimension (dimension calculation views)**
  As of SAP HANA SPS 10, we now use calculation views of type dimension, also called dimension calculation views. In SAP HANA SPS 11, a new migration tool is available to help migrate your old attribute views to the new dimension calculation views.

Just as the same master data is used in many cubes, so will we reuse dimension views in many of the other SAP HANA information views. You can even build your own “library” of dimension views.

Star Join Views

In the same way, instead of storing data in cubes, we can create another type of view in SAP HANA that produces the same results as a traditional cube. There are currently two types of SAP HANA views that produce the same results as an old-fashioned, traditional cube:

- **Analytic views**
  We still have an older version of these SAP HANA views called analytic views. This name refers to the fact that we use cubes for analytics. Analytic views are still used in some cases in SAP HANA SPS 11.

- **Calculation views of type cube with star join**
  As of SAP HANA SPS 10, we now use calculation views of type cube with star join. In SAP HANA SPS 11, a new migration tool is available to help you to migrate your old analytic views to the new calculation views of type cube with star join.

**Terminology**

In this book, we will simply refer to these types of views as star join views, not as cube views, which would lead to confusion. This is because the third type of SAP HANA views is called calculation view of type cube.
The results created by the star join views in SAP HANA are the same as that of a cube in an enterprise data warehouse, except that in a data warehouse, the data is stored in a cube. With SAP HANA, the data is not stored in a cube, but is calculated when required.

**Calculation Views of Type Cube**

This view type, provides even more powerful capabilities. For example, you can combine multiple star join views ("cubes") to produce a combined result set. In SAP Business Warehouse (BW), we call this combined result set a *MultiProvider*. In SAP HANA, we call these *calculation views of type cube*. Use cases for this type of view include:

- In your holding company, you want to combine the financial data from multiple subsidiary companies.
- You want to compare your actual expenses for 2015 with your planned budget for 2015.
- You have archived data in an archive database that you want to combine with current data in SAP HANA’s memory to compare sales periods over a number of years.

**Using Information Views**

Let’s examine how to use the different types of SAP HANA information views together in information modeling. Figure 4.11 provides an overview of everything you’ve learned in this chapter and how each topic fits together.

On the top left-hand side, you can see our source systems: an SAP source system and a non-SAP source system. SAP HANA doesn’t care from which system the data comes. In many projects, we have more non-SAP systems that we get data from than SAP systems. We extract the data from these systems using various data provisioning tools, which we’ll look at in Chapter 14.

The data is stored in row tables in these source systems. When we use our data provisioning methods and tools, the data is put into column tables in the SAP HANA database. This is illustrated in the bottom-left corner of Figure 4.11.

**Building the Information Model**

Now, let’s start building our information models on top of these tables.

If we want to build something like a cube for our analytics or reporting, we use SAP HANA information views. First, we need the equivalent of a dimension table, such as our Product or Customer dimension table. How would we build this?

We will build a dimension view (a calculation view of type dimension, or an attribute view). This is indicated in Figure 4.11 by Laura and Apples, which represent the Customer and Product dimension views. We create these dimension views the same way we create any other view: Take two tables, join them together, select certain output fields, and save it.
Once we have our dimension views, we build a data foundation for the cube. The data foundation is for all our transactional data. The data foundation stores measures, like the number 10 for the 10 apples that Laura bought. We build a data foundation in a similar fashion to building a dimension view: We take multiple tables or a single table. If using multiple tables, we join them together, select the output fields, and build our data foundation.

To complete the cube, we have to link our data foundation (fact tables) to the dimensions with a star join. Finally, to complete our star join view, we also select the output fields for our cube. The created star join view will produce the same result as a cube, but in this case no data is stored and it performs much better.

In the next step, we can create calculation view of type cube, in which we combine two star join views (cubes). We can combine the star join views using a join or a union. We recommend a union in this case, because it leads to better performance than a join.

Finally, we can build reports, analytics, or applications on our calculation view. This is shown in the top right of Figure 4.11.

Using the Information Model

In the previous section, we looked at how to build an information model from the bottom up. Let’s now think about the process from the other side and look at what happens when we call our report or application that built on the calculation view. This is the top-down view of Figure 4.11.

The calculation view does not store the data: It has to gather and calculate all the data required by the report. SAP HANA looks at the calculation view’s description and says, “Oh, I’ve got to build two star join views and union them together. How do I build these star join views?”

SAP HANA then goes down one level and says, “OK, how do I build the first star join view? I need a data foundation, and I need two dimension views, and I join them together with a star join.”

Then, it goes down another level: “How do I build the dimension view? I’ll read two different tables, join them together, and select the output fields.” SAP HANA then builds the two dimension views.

Going back up one level, SAP HANA needs to combine these two dimension tables with a data foundation. It creates the data foundation and joins it with the two dimension tables. It also builds the second star join view in the same way. Going up to the level of the calculation view, it combines the two star join views with a union. Finally, SAP HANA sends the result set out to the report, and then cleans out the memory again. (It does not store this data!) If someone else asks for the exact same data five minutes later, SAP HANA happily performs the same calculations all over again.

At first, this can seem very wasteful. Often we’re asked “Why don’t you use some kind of caching?” However, caching means that we store data in memory. In SAP HANA the entire database is already in memory. We don’t need caching because everything is in memory already.

The reason people ask this question is that in their traditional data warehouses, data normally doesn’t change until the next evening when the data is refreshed. Inside SAP HANA, the data can be updated continuously, and we always want to use the latest data in our applications, reports, and analytics. That means we have to recalculate the values each time; caching static result sets will not work.

In summary, to use real-time data, you have to continuously recalculate.

Other Modeling Artifacts

We will now discuss some of the other modeling concepts we will use when building our information views.

The power of calculation views of type cube is that you can build them up in many layers. In the bottom layer, you can perhaps have a union of two star join views (cubes), as shown in Figure 4.12. In the next (higher) layer, we sum up all the values from the union. In the level above that, we rank the summed results to show the top 10 most profitable customers.

Projection

A projection is used to change an output result set. For example, we can get a result set from a star join view (cube) and find that it has too many output fields for what we need in the next layers. Therefore, we use a projection as an additional in-between layer to create a subset of the results.
We can also use the projection to calculate additional fields—for example, sales tax—or we can rename fields to give them more meaningful names for our analytics and reporting users.

**Ranking**

As illustrated in the top node of Figure 4.12, we can use ranking to create useful top-N or bottom-N reports and analytics. This normally returns a filtered list (e.g., only the top 10 companies) and is sorted in the order we specify.

### Aggregation

The word *aggregation* means a collection of things. In SQL, we use aggregations to calculate values such as the sum of a list of values. Aggregations in databases provide the following functions:

- **Sum**
  Adds up all the values in the list
- **Count**
  Counts how many values (records) there are in the list
- **Minimum**
  Finds the smallest value in the list
- **Maximum**
  Finds the largest value in the list
- **Average**
  Calculates the arithmetical mean of all the values in the list
- **Standard deviation**
  Calculates the standard deviation of all the values in the list
- **Variance**
  Calculates the variance of all the values in the list

Normally, we would only calculate aggregations of measures. (Remember that measures are values we can perform calculations on, which usually means our transactional data.) For example, we could calculate the total number of apples a shop sold in 2015. However, in the latest versions of SAP HANA, we can also create aggregations of attributes, essentially creating a list of unique values.

**Union**

A union combines two result sets. Figure 4.12 illustrates this by combining the results of two star join views.

In SAP HANA graphical calculation views, we’re not restricted to combining two result sets; we can combine multiple result sets. SAP HANA merges these multiple result sets into a single result set.

Let’s say we have four result sets—A, B, C, and D—that we union together. Although we union multiple result sets, that doesn’t mean the performance will
be bad. If SAP HANA detects that you are not asking for data from A and D, it will not even generate the result sets for A and D; it will only work with B and C.

**Note**

The union expects the two result sets to have the same number of columns, in the same order, and the matching columns from each result set must have similar data types.

You can have variations in unions, such as the following:

- **Union all**
  Will merge the result sets and return all records

- **Union**
  Will merge the results and only give you the unique results back

- **Union with constant values**
  Lets you "pivot" your merged results to help get the data ready for reporting output

Figure 4.13 compares the output generated by a union to that of a union with constant values.

![Figure 4.13 Standard Union Compared to Union with Constant Values](image)

Often, we want a report that looks like the top right of the figure. If we're given the output from the standard union, it takes more work to create a report with sales for 2015 and 2016 in separate columns. With the output from the union with constant values, writing the report is easy.

In Figure 4.11, we showed that you should use a union rather than a join for performance reasons—but what happens if the two star join views that you want to union together do not have the same outputs? In that case, the union will return an error. How do we solve this problem?

Figure 4.14 illustrates an example. On top, we have the ideal case in which both star join views have matching columns. In the middle, we have a case in which only column B matches. Normally, we would use a join, as shown.

![Figure 4.14 Standard Union, Join, and Join Converted to Union](image)

We solve this problem by creating a projection for each of the two star join views. We add the missing columns and assign a NULL value to those columns. We then perform the union on the two projections, because they now have matching columns.
Semantics

An important step that many modelers skip is to add more meaning to the output of their information models. Semantics, as a general term, is the study of meaning. We can use semantics in SAP HANA to, for example, rename fields in a system so they are more clear. Some database fields have really horrible names. Using semantics, you can change the name of these fields to be more meaningful for your end users. This is especially useful when they create reports. If you have a reporting background, you will know that a reporting universe can fulfill a similar function.

We can also show that certain fields are related to each other—for example, a customer number and a customer name field.

One example that is used often is that of building a hierarchy. By using a hierarchy, you can see how fields are related to each other with regards to “drilling down” for reports.

Hierarchies

Hierarchies are used in reporting to enable intelligent drilldown. Figure 4.15 illustrates two examples: Users can start with a list of sales for all the different countries; they can then drill down to the sales of the provinces (or states) of a specific country, and finally down to the sales of a city.

End users can see the sales figures for just the country, state, or province or for a specific city. Even though countries and cities are two different fields, they’re related to each other by use of a hierarchy. This extra relationship enhances the data and gives it more meaning (semantic value).

There are two different types of hierarchies: level hierarchies and parent–child hierarchies. The hierarchy we just described is the level hierarchy; level 1 is the country, level 2 is the state or province, and level 3 is the city or town.

Another example of a level hierarchy is a time-based hierarchy that goes from a year to a month to a day. You can even keep going deeper, down to seconds. This is illustrated on the right side of Figure 4.15.

HR employee organizational charts or a cost center structures are examples of parent–child hierarchy, as illustrated in Figure 4.16. In this case, we join a table to itself, as described in the Self-Joins section.
Table 4.1 provides a quick summary of the differences between the two types of hierarchies and when you should use which one.

<table>
<thead>
<tr>
<th>Level Hierarchies</th>
<th>Parent–Child Hierarchies</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fixed (rigid) number of levels in the hierarchy.</td>
<td>Variable hierarchy depth.</td>
</tr>
<tr>
<td>Different data fields on every level of the hierarchy (e.g., country, state, city).</td>
<td>The same (two) fields used on every level of the hierarchy.</td>
</tr>
<tr>
<td>The fields on each level of the hierarchy can be different data types (e.g., country can be text, while zip/postal code can be numeric).</td>
<td>The parent and child fields have the same data type.</td>
</tr>
</tbody>
</table>

Table 4.1 Comparing Level Hierarchies and Parent–Child Hierarchies

**Best Practices and Modeling Guidelines**

Let's end our tour of modeling concepts with a summary of best practices for modeling views in SAP HANA. Many of the basic modeling guidelines that we find in traditional data modeling and reporting environments are still applicable in SAP HANA. We can summarize the basic principles as follows:

- **Limit (filter) the data as early as possible**
  
  It’s wasteful to send millions of data records across a slow network to a reporting tool when the report only uses a few of these data records. It makes a lot more sense to send only the data records that the report requires, so we want to filter the data before it hits the network.

  In the same way, we don’t want to send data to a star join view when we can filter it in the earlier dimension view. Hence, the rule to filter as early as possible.

- **Perform calculations as late as possible**
  
  Assume you have column A and column B in a table. The table contains billions of records. We can use the values of column A and column B in a formula for every data record, which means the server performs billions of calculations, and finally add up the sum total of the calculated fields.

  It would be much faster to add up the sum total for column A and the sum total for column B and then calculate the formula. In that case, we perform the calculation once instead of billions of times. We delay the calculation as long as possible for performance reasons.

**Important Terminology**

In this chapter, we focused on various modeling concepts and how they're used in SAP HANA. We introduced the following important terms:

- **Views**
  
  We started at by looking at views and realized that we can leverage the fact that SAP HANA already has all the data in memory, that the servers have lots of CPU cores available, and that SAP HANA runs everything in parallel. We therefore do not need to store the information in cubes and dimension tables, but can use views to generate the information when we need it. This also gives us the advantage of real-time results.

- **Joins**
  
  We have both normal database join types, like inner and left outer joins, and SAP HANA-specific types, like referential, text, temporal, and spatial joins. In the process, we also looked at the special case of self-joins and at dynamic joins as an optimization technique.

- **Core Data Services (CDS)**
  
  CDS provides a way to separate business semantics and intent from database operations.

- **Cubes**
  
  A cube consists of a data foundation with fact tables that contain the transactional data, linked to dimension tables that contain master data. The data foundation is linked to the dimension tables with star joins.

- **Attributes and measures**
  
  Attributes describe elements involved in a transaction. Transactional data that we can perform calculations on is called a measure and stored in the fact tables.

- **Information views**
  
  In SAP HANA, we take the concept of views to higher levels. We can create
dimension tables and cubes as dimension views and star join views. Dimension views are attribute views and calculation views of type dimension. Star join views are analytic views and calculation views of type cube with star join. The final type of information view is the calculation view of type cube, which we can use to perform even more powerful processes.

**Modeling artifacts**

Inside our views, we can use unions, projections, aggregations, and ranking. Unions are quite flexible and can be used for a direct merge of result sets, or we can use a union with constant values to create report-ready result sets.

**Practice Questions**

These practice questions will help you evaluate your understanding of the topics covered in this chapter. The questions shown are similar in nature to those found on the certification examination. Although none of these questions will be found on the exam itself, they will allow you to review your knowledge of the subject. Select the correct answers and then check the completeness of your answers in the Practice Question Answers and Explanations section. Remember that on the exam you must select all correct answers and only correct answers to receive credit for the question.

In this section, we have a few questions with graphics attached. In the certification exam, you might also see a few questions that include graphics.

1. In which SAP HANA views will you find measures? (There are 2 correct answers.)
   - [ ] A. Attribute view
   - [ ] B. Calculation view of type cube with star join
   - [ ] C. Calculation view of type cube
   - [ ] D. Database views

2. True or False: A database view stores data in the database.
   - [ ] A. True
   - [ ] B. False

3. A traditional cube is represented by which SAP HANA view type? (There are 2 correct answers.)
   - [ ] A. Attribute view
   - [ ] B. Analytic view
   - [ ] C. Calculation view of type cube with star join
   - [ ] D. Calculation view of type dimension

4. A referential join gives the same results as which other join type?
   - [ ] A. Inner join
   - [ ] B. Left outer join
   - [ ] C. Spatial join
   - [ ] D. Star join

5. Which join type makes use of date ranges?
   - [ ] A. Spatial join
   - [ ] B. Text join
   - [ ] C. Temporal join
   - [ ] D. Inner join

6. If we change the transaction “Laura buys 10 apples” to “Laura buys 10 green apples,” how would we store the color green?
   - [ ] A. Store green as an attribute
   - [ ] B. Store green as a measure
   - [ ] C. Store green as a spatial data type
   - [ ] D. Store green as a CDS view

7. True or False: A view always shows all the available columns of the underlying tables.
   - [ ] A. True
   - [ ] B. False
8. You have a view with two tables, joined by a left outer join. If you redesign the view and accidently swap the two tables around, what should you do to the join?

- A. Keep the left outer join.
- B. Change the join to a text join.
- C. Change the join to a right outer join.
- D. Change the join to a referential join.

9. What do you call the data displayed in the data foundation of an SAP HANA information view?

- A. Facets
- B. Measures
- C. Characteristics
- D. Key figures

10. You are writing a mobile application for the World Series. You have details about all the baseball players and the baseball scores for all the previous matches. How do you use the data of the player information and the scores?

- A. Both the player information and the scores are used as master data.
- B. Both the player information and the scores are used as transactional data.
- C. The player information is used as master data, and the scores are used as transactional data.
- D. The player information is used as transactional data, and the scores are used as master data.

11. Look at Figure 4.17. You are selling books that have been translated into various languages. What join type should you use?

- A. Left outer join
- B. Text join
- C. Temporal join
- D. Referential join

12. Which of the following are reasons you should use Core Data Services (CDS) to create a schema? (There are 2 correct answers.)

- A. The database administrator might type the name wrong.
- B. The database administrator should not get those privileges in the production system.
- C. The focus is on the business requirements.
- D. The focus is on the database administration requirements.

13. Look at Figure 4.18. A company sends out a lot of quotes. Some customers accept the quotes, and they’re invoiced. The company asks you to find a list of the customers that did NOT accept the quotes. How do you find the customers that received quotes but did NOT receive invoices?

- A. Use a right outer join.
- B. Use a left outer join.
Use a left outer join. Filter to show only the NULL values on the right table.

Use a right outer join. Filter to show only customers on the left table.

C. Referential join
D. Inner join

14. True or False: When you use a parent-child hierarchy, the depth in your hierarchy levels can vary.
   □ A. True
   □ B. False

15. Look at Figure 4.19. What type of join should you use? (*Hint: Watch out for distractors.*)
   □ A. Temporal join
   □ B. Text join

16. You have a list of map locations for clinics. The government wants to build a new clinic, but wants to build it where there is the greatest need. You need to find the largest distance between any two clinics. With your current knowledge, how do you do this?
   □ A. Use a temporal join to find the distance between clinics.
       Use a union with constant values to “pivot” the values.
   □ B. Use a spatial join to find the distance between clinics.
       Use a level hierarchy for drilldown.
Use a dynamic join to find the longest distance between clinics.
Use a ranking to find the top 10 values.
Use a spatial join to find the distance between clinics.
Use an aggregation with the maximum option.

17. Look at Figure 4.20. What are the values for X, Y, and Z? (Note: You will not
see a question like this on the certification exam; it’s only added here to test
your understanding.)
A. X = Customer 2, Y = 400, Z = 800.
B. X = Customer 1, Y = 400, Z = 400.
C. X = Customer 1, Y = 800, Z = 800.
D. X = Customer 2, Y = 200, Z = 400.

18. You have to build a parent–child hierarchy. What type of join do you expect
to use?
A. Relational join
B. Temporal join
C. Dynamic join
D. Self-join

19. Look at Figure 4.21. What join type should you use to see all the suppliers?
A. Left outer join
B. Right outer join
C. Inner join
D. Referential join

---

**Figure 4.20** Find Values for X, Y, and Z

**Figure 4.21** What Join Should You Use to See All Suppliers?
Practice Question Answers and Explanations

1. Correct answers: B, C
   You find measures in analytic views, calculation views of type cube with star
   join (the new type of analytic view), and calculation views of type cube.
   Attribute views do not contain measures—only attributes!

2. Correct answer: B
   False. A database view does not store data in the database.

3. Correct answers: B, C
   A traditional cube is represented by an analytic view or a calculation view of
   type cube with star join. Both are different from a cube in that they do not
   store any data like a cube does. An attribute view and a calculation view of
   type dimension are similar to a dimension table.

4. Correct answer: A
   A referential join gives the same results as an inner join but speeds up the
   calculations in some cases by assuming referential integrity of the data. It is
   the optimal join type in SAP HANA.

5. Correct answer: C
   A temporal join requires FROM and TO fields in the table on the right side of
   the join and a date-time column in the table on the left side of the join.

6. Correct answer: A
   We cannot "measure" or perform calculations on the color green. Green is
   not a spatial data type. CDS views are not relevant in this context.

7. Correct answer: B
   False. A view does NOT always show all the available columns of the under-
   lying tables. You have to select the output fields of a view. The word always
   in the statement is what makes it false.

8. Correct answer: C
   A right outer join is the inverse of the left outer join. Therefore, if you
   reverse the order of the tables, you can "reverse" the join type. It is impor-
   tant to note that this does not work for a text join. A text join is only equi-
   valent to a left outer join. For a text join, you have to be careful about the
   order of the tables.

9. Correct answer: B
   The data displayed in the data foundation of an SAP HANA information view
   is called measures. In SAP HANA, we talk about attributes and measures. The
   other names might be used in other data modeling environments, but not in
   SAP HANA.

10. Correct answer: C
    The player information is used as master data, and the scores are used as
    transactional data. You cannot perform calculations on the players. There-
    fore, this data is used in the dimensions and is seen as master data. You def-
    initely perform calculations on the scores, so these are used in the fact tables,
    which means they are transactional data.

11. Correct answer: B
    Figure 4.17 shows the SPRAS column and shows a list of languages. The need
    for a translation is also hinted at in the question, meaning this should be a
    text join. There is not enough information to select any of the other answer
    options.

12. Correct answers: B, C
    You should use CDS to create a schema because the database administrator
    should not get those privileges in the production system and the focus
    should be on the business requirements.

13. Correct answer: C
    Look carefully at Figure 4.18. The question asks you to find all the customers
    that received quotes but did NOT receive invoices.
    The answer only gives us left outer or right outer as options. Because the
    question states they want all customers from the quotes tables, you might try
    a left outer join first.
    Now look at Figure 4.3 to see the difference between the left and right outer
    joins. Look especially at the where the NULL values are shown. With a left
    outer join, the NULL values are found in the right table. With a right outer
    join, the NULL values are found in the left table.
    We want to find the customers that did not get invoices. With a left outer
    join, the NULL values are found on the right side: That group is the one we
    are looking for. Therefore, filter for the NULL values on the right table.
Note

Any time that a negative phrase or word is used in the certification exam, it is shown in capital letters to bring attention to the fact that it is a negative word.

14. Correct answer: A
True. Yes, the depth of your hierarchy levels can vary when you use a parent-child hierarchy.

15. Correct answer: C
You should use a referential join.
There are two distractors to try and throw you off track:
▶ A temporal join uses FROM and TO fields, but only for date-time fields. This example is for airports.
▶ The second distractor was to try to get you to select the text join: The language code has nothing to do with translations, but has to do with the airline’s primary language.

Choose a referential join because you can see the same people traveling in both tables, so it seems that these two tables have referential integrity.

If the tables did not have referential integrity, you would have chosen an inner join.

16. Correct answer: D
A spatial join can give the distance between different points on a map.
There are two ways you can find the largest distance between any two clinics:
▶ Use a ranking to find the top 10 values.
▶ Use an aggregation with the maximum option.

17. Correct answer: B
See the full solution in Figure 4.22.
X = Customer 1, Y = 400, Z = 400.
Note that we are using average in the aggregation node, not sum.
In the ranking node, we are asking for the bottom 10, not the top 10.

18. Correct answer: D
You expect to use a self-join when you build a parent-child hierarchy. See Figure 4.16 for an illustration.

19. Correct answer: B
To see everything on the right table, use a right outer join.
Note that these tables do not have referential integrity. If we had asked the question a little differently, this could have been important.

**Takeaway**

You should now have a general understanding of information modeling concepts: views, joins, cubes, star joins, data foundations, dimension tables, and so on. In this chapter, you learned about the differences between attributes and measures and the value of CDS in the development cycle. You have seen the various types of views that SAP HANA uses, how to use them together, and what options you have available for your information modeling. Finally, we examined a few best practices and guidelines for modeling in SAP HANA in general.
Summary

You have learned about many modeling concepts and how they are applied and implemented in SAP HANA.

You’re now ready to go into the practical details of learning how to use all the SAP HANA information views and how to enhance these views with the various modeling artifacts that we have available to us in each of these different views.
Contents

Acknowledgments .......................................................................................... 15
Preface ........................................................................................................... 17

1 SAP HANA Certification Track—Overview ........................................... 23
   Who This Book Is For ................................................................................ 24
   SAP HANA Certifications ....................................................................... 25
      Associate-Level Certification ............................................................. 27
      Professional-Level Certification ....................................................... 28
      Specialist-Level Certification ........................................................... 28
   SAP HANA Application Associate Certification Exam ......................... 29
      Exam Objective .................................................................................. 31
      Exam Structure .................................................................................. 31
      Exam Process .................................................................................... 33
   Summary ................................................................................................... 34

2 SAP HANA Training .................................................................................... 35
   SAP Education Training Courses .......................................................... 36
      Training Courses for SAP HANA Certifications ............................... 37
      Additional SAP HANA Training Courses ......................................... 38
   Other Sources of Information ............................................................... 39
      SAP Help ............................................................................................ 39
      hana.sap.com and SAP Community Network ..................................... 41
      SAP HANA Academy ....................................................................... 41
      openSAP and openHPI .................................................................... 44
   Hands-On with SAP HANA .................................................................... 46
      Where to Get an SAP HANA System ............................................... 46
      Project Examples .............................................................................. 52
      Where to Get Data ............................................................................ 54
   Exam Questions ....................................................................................... 56
      Types of Questions ............................................................................. 56
      Elimination Technique ....................................................................... 58
      Bookmark Questions .......................................................................... 59
   General Examination Strategies ........................................................... 59
   Summary ................................................................................................... 60
### 3. Architecture and Deployment Scenarios ............................................ 61
- Objectives of This Portion of the Test .................................................. 62
- Key Concepts Refresher ................................................................. 63
- In-Memory Technology ................................................................. 63
- Architecture and Approach ........................................................... 69
- Deployment Scenarios ................................................................. 79
- Important Terminology ............................................................. 89
- Practice Questions ................................................................. 91
- Practice Question Answers and Explanations ........................................ 96
- Takeaway .................................................................................. 99
- Summary .................................................................................. 99

### 4. Information Modeling Concepts ......................................................... 101
- Objectives of This Portion of the Test .................................................. 102
- Key Concepts Refresher ................................................................. 103
- Tables ....................................................................................... 103
- Views ......................................................................................... 103
- Cardinality ............................................................................... 106
- Joins ......................................................................................... 107
- Core Data Services Views ........................................................... 117
- Cube ......................................................................................... 119
- Information Views ...................................................................... 122
- Using Information Views ........................................................... 124
- Other Modeling Artifacts ............................................................. 127
- Semantics ................................................................................ 132
- Hierarchies ............................................................................. 132
- Best Practices and Modeling Guidelines ........................................... 134
- Important Terminology ............................................................. 135
- Practice Questions .................................................................... 136
- Practice Question Answers and Explanations ........................................ 144
- Takeaway .................................................................................. 147
- Summary .................................................................................. 148

### 5. Information Modeling Tools ................................................................. 149
- Objectives of This Portion of the Test .................................................. 150
- Key Concepts Refresher ................................................................. 151
- SAP HANA Studio ........................................................................ 151

### 6. Information Views ............................................................................. 181
- Objectives of This Portion of the Test .................................................. 182
- Key Concepts Refresher ................................................................. 183
- Data Sources for Information Views ................................................... 184
- Calculation Views: Type Dimension, Type Cube, and Type Cube with Star Join .................................................. 185
- Working with Nodes ................................................................. 197
- Semantics Node ........................................................................ 202
- Attribute Views and Calculation Views of Type Dimension .................. 205
- Analytic Views and Calculation Views of Type Cube with Star Join .................. 208
- Migrating Attribute and Analytic Views ........................................... 209
- Important Terminology ............................................................. 213
- Practice Questions .................................................................... 214
- Practice Question Answers and Explanations ........................................ 218
- Takeaway .................................................................................. 220
- Summary .................................................................................. 221

### 7. Advanced Information Modeling ......................................................... 223
- Objectives of this Portion of the Test .................................................. 224
- Key Concepts Refresher ................................................................. 225
- Calculated Columns ................................................................. 225
- Restricted Columns ................................................................. 231
- Filters ......................................................................................... 236
- Variables ................................................................................ 238
- Input Parameters ..................................................................... 242
- Currency ................................................................................. 246
- Decision Tables ....................................................................... 250
- Hierarchies ............................................................................. 253
- Important Terminology ............................................................. 258
- Practice Questions .................................................................... 260
- Practice Question Answers and Explanations ........................................ 264
<table>
<thead>
<tr>
<th>Chapter</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>13</td>
<td>Security</td>
<td>433</td>
</tr>
<tr>
<td></td>
<td>Objectives of This Portion of the Test</td>
<td>434</td>
</tr>
<tr>
<td></td>
<td>Key Concepts Refresher</td>
<td>435</td>
</tr>
<tr>
<td></td>
<td>Usage and Concepts</td>
<td>435</td>
</tr>
<tr>
<td></td>
<td>Users</td>
<td>439</td>
</tr>
<tr>
<td></td>
<td>Roles</td>
<td>443</td>
</tr>
<tr>
<td></td>
<td>Privileges</td>
<td>446</td>
</tr>
<tr>
<td></td>
<td>Testing Security</td>
<td>458</td>
</tr>
<tr>
<td></td>
<td>Important Terminology</td>
<td>459</td>
</tr>
<tr>
<td></td>
<td>Practice Questions</td>
<td>461</td>
</tr>
<tr>
<td></td>
<td>Practice Question Answers and Explanations</td>
<td>463</td>
</tr>
<tr>
<td></td>
<td>Takeaway</td>
<td>465</td>
</tr>
<tr>
<td></td>
<td>Summary</td>
<td>465</td>
</tr>
<tr>
<td>14</td>
<td>Data Provisioning</td>
<td>467</td>
</tr>
<tr>
<td></td>
<td>Objectives of This Portion of the Test</td>
<td>468</td>
</tr>
<tr>
<td></td>
<td>Key Concepts Refresher</td>
<td>469</td>
</tr>
<tr>
<td></td>
<td>Concepts</td>
<td>470</td>
</tr>
<tr>
<td></td>
<td>SAP Data Services</td>
<td>476</td>
</tr>
<tr>
<td></td>
<td>SAP LT Replication Server</td>
<td>478</td>
</tr>
<tr>
<td></td>
<td>SAP Replication Server</td>
<td>484</td>
</tr>
<tr>
<td></td>
<td>SAP Direct Extractor Connection</td>
<td>485</td>
</tr>
<tr>
<td></td>
<td>SAP HANA Smart Data Access</td>
<td>488</td>
</tr>
<tr>
<td></td>
<td>SAP HANA Enterprise Information Management</td>
<td>495</td>
</tr>
<tr>
<td></td>
<td>SAP HANA Smart Data Streaming</td>
<td>497</td>
</tr>
<tr>
<td></td>
<td>Flat Files or Microsoft Excel Datasheets</td>
<td>500</td>
</tr>
<tr>
<td></td>
<td>Web Services (ODATA and REST)</td>
<td>502</td>
</tr>
<tr>
<td></td>
<td>Important Terminology</td>
<td>502</td>
</tr>
<tr>
<td></td>
<td>Practice Questions</td>
<td>505</td>
</tr>
<tr>
<td></td>
<td>Practice Question Answers and Explanations</td>
<td>508</td>
</tr>
<tr>
<td></td>
<td>Takeaway</td>
<td>510</td>
</tr>
<tr>
<td></td>
<td>Summary</td>
<td>511</td>
</tr>
<tr>
<td>15</td>
<td>Utilization of Information Models</td>
<td>513</td>
</tr>
<tr>
<td></td>
<td>Objectives of this Portion of the Test</td>
<td>514</td>
</tr>
<tr>
<td></td>
<td>Key Concepts Refresher</td>
<td>515</td>
</tr>
<tr>
<td></td>
<td>Business Intelligence Concepts</td>
<td>515</td>
</tr>
</tbody>
</table>
Index

A

ABAP, 477
ABC analysis algorithm, 329
Accelerator deployment, 84
profitability analysis, 84
ACID-compliant database, 75
Administration, 367, 368
Administration Console, 152, 163, 343, 352, 360, 365
diagnosis files, 163
performance, 163
Affinity propagation, 328
Aggregation node, 197, 213, 216, 219
calculated columns, 226
restricted columns, 232
Aggregations, 129, 136, 344
Algorithms, 327
Amazon Web Services (AWS), 47, 49, 88, 94
American National Standards Institute (ANSI), 271
Analytic privileges, 449, 460, 465
assign to role, 450
create, 450
migration, 455, 463
Analytic views, 123, 136, 137, 144, 182, 208, 215
calculate before aggregation, 229
create, 186
migration, 209
temporal joins, 220
vs. calculation views of type cube with star join, 208
Analytics, 159, 517, 538
API, 318, 333
SAP HANA simple info access (SINA), 318
SAP HANA Text Analysis XS JavaScript, 322
SAP HANA Text Mining XS JavaScript, 322
Application Function Library (AFL), 278, 329
Application Function Modeler (AFM), 179, 330, 331, 332, 336
deployment, 168
PAI, 168

Applications, 344
development, 167
privileges, 457
Apriori algorithm, 328
Architects, 25
Architecture, 69, 90, 343, 359
Association algorithms, 328
Asynchronous replication, 79
Attribute views, 123, 136, 144, 182, 205, 217
create, 186
derived, 207
migration, 185, 209
vs. calculation view of type dimension, 205
Attributes, 101, 120, 121, 135
calculated columns, 228
graphic, 254
restricted columns, 232
restrictions, 453
vocabulary, 252
Authentications, 441
Authorization Assistant, 436
Authorizations, 359, 441, 442
Auto Documentation, 387, 392, 397
Automated predictive library, 531
Average, 129

B

Backups, 76, 154
Best practices, 134
BEx Analyzer, 522
Big data, 493
limitations, 493
BIGINT, 307
Bookmark questions, 59
Bottlenecks, 69
Branch logic, 359
Bring-your-own-license, 89
Business Explorer (BEx), 519
Business Function Library (BFL), 329
Business information, 513
Index

Business intelligence, 513, 514
applications and dashboards, 527
choosing the right tool, 533
consuming information models, 515
convergence of tools, 517
data discovery, 529
fact sheet, 523
history, 516
reporting, 532
tools, 519
Business Intelligence Consumer Services (BICS), 476, 504, 526, 540
Business rules, 250

C
C_HANAIMP, 27
C_HANAIMP_11, 17, 24
scoring, 33
C_HANATEC, 27, 29
C4.5 decision tree, 331
Caching, 127
core data, 79
column, 103
calculated columns, 102, 223, 225, 226, 258,
columnar storage, 344, 361
column-based
views, 156
correlation, 183, 214, 216
create, 226
columnar views, 244
calculating engine, 346, 365
calculations, 126, 185, 263, 346, 536
adding data sources, 192
adding nodes, 191
create, 186
creating joins, 193
default nodes, 190
output area, 246
save and activate, 196
selecting output fields, 194
variables, 241
Calculations, 134, 345
CALL statement, 290
Cardinality, 106, 193, 356, 358, 361
many-to-many, 107
many-to-one, 107
one-to-many, 106
types, 106
Catalog, 178
SAP HANA studio, 155, 156, 176
SAP HANA web-based development
workbench, 171
table context menu, 156
table preview, 172
tables, 156
CDX views, 117, 135, 139
ABAP, 118
benefits, 117
data sources, 118
Certification
prefixes, 26
track, 23
Change and Transport System (CTS), 376
Characteristics, 120
Circularstring, 324
Classical analytic privileges, 204
Classification algorithms, 328
Classroom training, 36
Clients, 168
Client-side JavaScript, 86
Cloud, 170
deployments, 87
Clustering algorithms, 327, 329
cold data, 79
column, 103
gen, 346
Column (Cont.)
store, 156
table, 78, 157, 272
views, 156
Column Views folder, 373
Columnar storage, 344, 361
Column-based
storages, 73
inputs, 243, 259
storage, 73
table, 68, 156, 176
Column-oriented tables, 103
Comments, 387
Complex event processing, 517, 538
Computation Continuous Language (CCL), 499
Conditional statements, 278
Consumers, 520
Consumption, 534
Contains predicate, 317
Current, 159
create modeling object, 162
packages, 159
SAP HANA studio, 155, 159
SHINE package, 161
subpackages, 159
Context menu, 156
Controllers, 535
Convergence, 517
Core Data Services (CDS), 101, 117, 145,
184, 213, 315, 383, 393
Count, 129
Counters, 230, 258
new, 231
CPU
in parallel, 71
speed, 64
Create Fulltext Index, 314
Creators, 520
CSS, 86, 151
CSV files, 500
Cubes, 101, 119, 135, 137, 144
Currency, 246, 248
conversion, 247, 248
Currency (Cont.)
conversion options, 249
conversion schema, 249
decimal shift, 248
decimal shift back, 249
source and target, 247
Current conversions, 102, 223, 259, 261,
278, 299
Customer query views, 411, 424
D
Dashboards, 534
Data
aging, 72, 79
archiving, 494
backups, 76
category, 187, 188
correlation, 276
deployments, 87
data discovery, 182, 126, 135, 145, 251, 345
data foundation, 25
data lakes, 493
noise, 328
persistence, 77
provisioning specialist, 25
read, 276, 361
records, 272
scientists, 25
sources, 184, 213
streaming, 497
types, 273
volume, 76, 77, 78, 79, 99
Data Control Language (DCL), 273, 306
Data Definition Language (DDL), 273,
284, 306
data foundation node, 215
data manipulation node, 215
data transformation node, 215
data modeling
artifacts, 127, 136
concepts, 101
limit and filter, 134
data preview, 157
analysis, 159
Dashes, 534
Data
aging, 72, 79
archiving, 494
backups, 76
category, 187, 188
correlation, 276
deployments, 87
data discovery, 182, 126, 135, 145, 251, 345
data foundation, 25
data lakes, 493
noise, 328
persistence, 77
provisioning specialist, 25
read, 276, 361
records, 272
scientists, 25
sources, 184, 213
streaming, 497
types, 273
volume, 76, 77, 78, 79, 99
Data Control Language (DCL), 273, 306
Data Definition Language (DDL), 273,
284, 306
data foundation node, 215
data manipulation node, 215
data transformation node, 215
data modeling
artifacts, 127, 136
concepts, 101
limit and filter, 134
data preview, 157
analysis, 159
Roles (Cont.)
template roles, 443, 444
Root package privilege, 457
Row, 103
engine, 346
storage, 344
store, 156
Row-based
databases, 73
storage, 73
tables, 68, 156, 176
Row-oriented tables, 103
Ruby, 151
Runtime information, 157
Runtime version, 373, 390
Ruby, 151
Runtime roles, 445
S
Sales forecasting, 327
Sandbox, 500
SAP Basis, 27
SAP Business Suite, 479
SAP Business Warehouse (BW), 82
SAP BusinessObjects Analysis, edition for
Microsoft Office, 519, 522, 525, 539, 541
information, 526
plug-in, 526
SAP BusinessObjects BI Mobile, 529
SAP BusinessObjects Design Studio, 519,
521, 527, 529, 539, 540
dashboard, 528
information, 527
SAP BusinessObjects Web Intelligence,
519, 532
information, 532
SAP BW, 121, 156, 435, 475, 486, 517, 536
consume calculation views, 537
consume information models, 537, 541
generate SAP HANA views, 537
SAP BW on SAP HANA, 38, 51, 82, 156, 464
SAP Cloud Appliance Library, 49, 50
SAP Cloud for Analytics, 522, 529, 540
SAP Community Network (SCN), 41
SAP CRM, 435
SAP Crystal Reports, 519, 532
SAP Crystal Reports 2011/2013, 533, 539
SAP Crystal Reports for Enterprise, 533,
539, 540
information, 533
SAP Data Quality Management, 478
SAP Data Services, 476, 505
benefits, 478
extracting data, 477
transforming and cleaning data, 478
SAP Direct Extractor Connection (D XC), 477,
485, 487
SAP Education, 35
SAP ERP, 408, 435
SAP ERP on SAP HANA, 156
SAP extractors, 474, 503
SAP Fiori, 86
SAP HANA
as a database, 82, 83
as a development platform, 84, 85
as a sidescar solution, 80, 81
as a virtual machine, 87
as an accelerator, 84
clients, 475
in the cloud, 87
reference guides, 40
training courses, 35, 37
SAP HANA Academy, 41
SAP HANA application function modeler
(AF M), 168
SAP HANA Application Lifecycle Manager,
329, 378, 391
SAP HANA as a database, 409
security, 435
SAP HANA as a platform, 90
security, 436, 461
SAP HANA as a sidescar, 409
security, 436
SAP HANA Business Function Library (BFL)
Reference, 41
SAP HANA certifications
associate level, 27
professional level, 28
SAP HANA Cloud Platform, 47, 48, 87,
89, 522
SAP HANA Developer Center, 46
SAP HANA Developer Guide, 41
SAP HANA Developer Quick Start Guide, 41
SAP HANA Enterprise Cloud (HEC), 88, 89,
91, 94
private cloud, 89
SAP HANA Enterprise Information Management, 495
SAP HANA Interactive Education (SHINE),
39, 530
SAP HANA Live, 364, 403, 404, 461, 464
architecture, 410
background information, 405
definition, 405
installation, 413, 425
rapid deployment solution, 415
security, 436
SQL engine, 346
tags, 419
views, 403, 412, 422, 424
SAP HANA Live Browser, 403, 414, 415, 425
all views, 416
for business users, 421, 430
invalid views, 416
my favorites, 416
toolbar, 418
SAP HANA Live Extension Assistant, 403,
414, 425, 426
restrictions, 423
SAP HANA Modeling Guide, 39
SAP HANA Predictive Analysis Library (PAL)
Reference, 41
SAP HANA Security Guide, 40
SAP HANA simple info access (SINA) API,
318, 333
SAP HANA smart data access (SDA), 79,
356, 488
adapters, 488
benefits, 495
data archiving, 494
implementation, 490
virtual tables, 488
SAP HANA smart data integration (SDI), 168,
495, 496
SAP HANA smart data quality (SDQ),
495, 496
SAP HANA smart data streaming (SDS), 168,
497, 498, 517
benefits, 499
SAP HANA SPS 10, 322
SAP HANA SPS 11, 283, 295, 319, 455
SAP HANA SQLScript Reference, 40
SAP HANA Studio, 149, 150, 151, 175,
445, 514
Administration Console, 163
Backup, 154
Catalog, 155
Content, 155, 159
create calculation view, 187
data preview, 534
development object, 167
installation, 52
main workspace, 154
perspectives, 152
Provisioning, 155
Quick View tab, 152
screen, 152
security, 156
session client, 152, 168
SQL Console, 162
systems view, 154
working areas, 152
XS project, 166
SAP HANA Text Analysis XS JavaScript
API, 322
SAP HANA Troubleshooting and Performance
Analysis Guide, 40
SAP HANA web-based development
workbench, 52, 150, 170, 175, 414, 445
table data, 174
SAP HANA XS, 85, 86, 179, 291, 317,
535, 541
application privileges, 457, 460
project, 166
SAP HANA XS DB Utilities JavaScript API
Reference, 41
SAP HANA XS JavaScript API Reference, 41
SAP HANA XS JavaScript Reference, 40
SAP HANA XSUnit JavaScript API
Reference, 41
SAP Help, 39
SAP Help, 39
SAP HANA smart data streaming (SDS), 168,
497, 498, 517
benefits, 499
SAP HANA SPS 10, 322
SAP HANA SPS 11, 283, 295, 319, 455
SAP HANA SQLScript Reference, 40
SAP HANA Studio, 149, 150, 151, 175,
445, 514
Administration Console, 163
Backup, 154
Catalog, 155
Content, 155, 159
create calculation view, 187
data preview, 534
development object, 167
installation, 52
main workspace, 154
perspectives, 152
Provisioning, 155
Quick View tab, 152
screen, 152
security, 156
session client, 152, 168
SQL Console, 162
systems view, 154
working areas, 152
XS project, 166
SAP HANA Text Analysis XS JavaScript
API, 322
SAP HANA Troubleshooting and Performance
Analysis Guide, 40
SAP HANA web-based development
workbench, 52, 150, 170, 175, 414, 445
table data, 174
SAP HANA XS, 85, 86, 179, 291, 317,
535, 541
application privileges, 457, 460
project, 166
SAP HANA XS DB Utilities JavaScript API
Reference, 41
SAP HANA XS JavaScript API Reference, 41
SAP HANA XS JavaScript Reference, 40
SAP HANA XSUnit JavaScript API
Reference, 41
SAP Help, 39
SAP Help, 39
<table>
<thead>
<tr>
<th>Page</th>
<th>Section</th>
<th>Text</th>
</tr>
</thead>
<tbody>
<tr>
<td>556</td>
<td>Index</td>
<td>SAP Hybris, 88</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SAP InfiniteInsight, 530</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SAP landscape, 375</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SAP Learning Hub, 37</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SAP LT Replication Server (SLD), 409, 478</td>
</tr>
<tr>
<td></td>
<td></td>
<td>benefits, 483</td>
</tr>
<tr>
<td></td>
<td></td>
<td>configuration, 482</td>
</tr>
<tr>
<td></td>
<td></td>
<td>features, 481</td>
</tr>
<tr>
<td></td>
<td></td>
<td>replication process, 480</td>
</tr>
<tr>
<td></td>
<td></td>
<td>trigger-based replication, 480</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SAP Lumira, 54, 421, 519, 521, 529, 531, 540</td>
</tr>
<tr>
<td></td>
<td></td>
<td>charts, 530</td>
</tr>
<tr>
<td></td>
<td></td>
<td>information, 529</td>
</tr>
<tr>
<td></td>
<td></td>
<td>storyboard, 531</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SAP NetWeaver old architecture, 65</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SAP Predictive Analytics, 529, 530, 540</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SAP Replication Server (SRS), 484</td>
</tr>
<tr>
<td></td>
<td></td>
<td>benefits, 485</td>
</tr>
<tr>
<td></td>
<td></td>
<td>log-based replication, 484</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SAP resources, 39</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SAP S/4HANA, 72, 80, 86, 97, 464</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SAP Solution Manager, 376</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SAP Store, 53</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SAP SuccessFactors, 88</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SAP Support Mode, 377, 391</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SAPUI5, 25, 38, 41, 85, 86</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Developer Guide for SAP HANA, 41, views, 535</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Savepoint, 76, 78, 99</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Scalable Vector Graphic (SVG), 326</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Scalar functions, 269, 283, 285</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Scale-out architecture, 75</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Schema, 156, 162, 394, 396</td>
</tr>
<tr>
<td></td>
<td></td>
<td>mapping, 367, 368, 380, 393</td>
</tr>
<tr>
<td></td>
<td></td>
<td>tables, 156</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SCORE() function, 317</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Scripted calculation views, migrate, 297</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Search, 314</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Seasonal patterns, 327</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Secondary storage, 76</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Security, 282, 359, 433, 438</td>
</tr>
<tr>
<td></td>
<td></td>
<td>concepts, 435, 437</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SAP HANA studio, 156</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Security (Cont.)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SAP HANA web-based development workbench, 171</td>
</tr>
<tr>
<td></td>
<td></td>
<td>testing, 458</td>
</tr>
<tr>
<td></td>
<td></td>
<td>usage, 438</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Segmentation, 328</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SELECT *, 303, 306</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SELECT statement, 276</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Self-joins, 110, 135</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Self-service, 520, 523, 534</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Semantics node, 202, 208, 214, 215</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Column tab, 202</td>
</tr>
<tr>
<td></td>
<td></td>
<td>hide fields, 203</td>
</tr>
<tr>
<td></td>
<td></td>
<td>hierarchies, 254</td>
</tr>
<tr>
<td></td>
<td></td>
<td>input parameters, 242</td>
</tr>
<tr>
<td></td>
<td></td>
<td>renaming fields, 202</td>
</tr>
<tr>
<td></td>
<td></td>
<td>session client, 204</td>
</tr>
<tr>
<td></td>
<td></td>
<td>top node, 189</td>
</tr>
<tr>
<td></td>
<td></td>
<td>variables, 239</td>
</tr>
<tr>
<td></td>
<td></td>
<td>View Properties tab, 203</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Sentiment analysis, 320, 334</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Separate statements, 279</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Sequential execution, 289</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Servers, 75</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Session client, 152, 168, 440</td>
</tr>
<tr>
<td></td>
<td></td>
<td>settings, 169</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Set-oriented, 272, 300</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Shared hardware, 79</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SHINE, 53, 118, 160, 161, 287, 326, 347</td>
</tr>
<tr>
<td></td>
<td></td>
<td>datasets, 54</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Show Line Numbers, 162</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Sidecar deployment, 80</td>
</tr>
<tr>
<td></td>
<td></td>
<td>advantages, 81</td>
</tr>
<tr>
<td></td>
<td></td>
<td>blank system, 82</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SINA, 318, 333</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Single sign-on, 441, 462</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Single value, 239, 258</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Slow disk, 67</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Social network analysis algorithms, 329</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Software, 51</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Software as a Service (SaaS), 88</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Spatial (Cont.)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>join type, 278</td>
</tr>
<tr>
<td></td>
<td></td>
<td>processing, 311, 323, 333</td>
</tr>
<tr>
<td></td>
<td></td>
<td>properties, 324</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SQL, 85, 269, 272, 341</td>
</tr>
<tr>
<td></td>
<td></td>
<td>analytic privileges, 204, 452, 455, 462, 463</td>
</tr>
<tr>
<td></td>
<td></td>
<td>button, 162</td>
</tr>
<tr>
<td></td>
<td></td>
<td>conditional statements, 278</td>
</tr>
<tr>
<td></td>
<td></td>
<td>creating calculated columns, 276</td>
</tr>
<tr>
<td></td>
<td></td>
<td>creating projections, 276</td>
</tr>
<tr>
<td></td>
<td></td>
<td>creating tables, 275</td>
</tr>
<tr>
<td></td>
<td></td>
<td>creating unions, 277</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Data Definition Language, 273</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Data Manipulation Language, 273</td>
</tr>
<tr>
<td></td>
<td></td>
<td>dynamic, 282, 302, 307</td>
</tr>
<tr>
<td></td>
<td></td>
<td>engine, 204, 346, 413, 422, 425</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Expression Editor, 228</td>
</tr>
<tr>
<td></td>
<td></td>
<td>filter data, 276</td>
</tr>
<tr>
<td></td>
<td></td>
<td>guidelines, 359</td>
</tr>
<tr>
<td></td>
<td></td>
<td>language, 272</td>
</tr>
<tr>
<td></td>
<td></td>
<td>reading data, 276</td>
</tr>
<tr>
<td></td>
<td></td>
<td>security, 284, 289</td>
</tr>
<tr>
<td></td>
<td></td>
<td>set-oriented, 272</td>
</tr>
<tr>
<td></td>
<td></td>
<td>statements, 162</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Structured Query Language, 271, 299</td>
</tr>
<tr>
<td></td>
<td></td>
<td>views, 105, 184, 213</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SQL Console, 152, 162, 290, 347, 353</td>
</tr>
<tr>
<td></td>
<td></td>
<td>schemas, 162</td>
</tr>
<tr>
<td></td>
<td></td>
<td>text editors, 162</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SQL Editor, 454</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SQL Plan Cache, 365</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SQLScript, 85, 269, 272, 278, 292, 307, 331, 341, 359, 362, 384</td>
</tr>
<tr>
<td></td>
<td></td>
<td>compiler, 279</td>
</tr>
<tr>
<td></td>
<td></td>
<td>decision tables, 253</td>
</tr>
<tr>
<td></td>
<td></td>
<td>declarative logic, 281</td>
</tr>
<tr>
<td></td>
<td></td>
<td>dynamic SQL, 282</td>
</tr>
<tr>
<td></td>
<td></td>
<td>for loop, 281</td>
</tr>
<tr>
<td></td>
<td></td>
<td>multilevel aggregation, 292</td>
</tr>
<tr>
<td></td>
<td></td>
<td>optimizer, 279</td>
</tr>
<tr>
<td></td>
<td></td>
<td>procedures, 288</td>
</tr>
<tr>
<td></td>
<td></td>
<td>security, 282, 283</td>
</tr>
<tr>
<td></td>
<td></td>
<td>separate statements, 279</td>
</tr>
<tr>
<td></td>
<td></td>
<td>while loop, 281</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Standard deviation, 129</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Tables, 103</td>
</tr>
<tr>
<td></td>
<td></td>
<td>context menu, 176</td>
</tr>
<tr>
<td></td>
<td></td>
<td>create, 275</td>
</tr>
<tr>
<td></td>
<td></td>
<td>data preview, 157</td>
</tr>
<tr>
<td></td>
<td></td>
<td>data source, 184, 213</td>
</tr>
<tr>
<td></td>
<td></td>
<td>definitions, 156</td>
</tr>
<tr>
<td></td>
<td></td>
<td>export data, 378</td>
</tr>
<tr>
<td></td>
<td></td>
<td>functions, 251, 269, 283, 286</td>
</tr>
<tr>
<td></td>
<td></td>
<td>join, 345</td>
</tr>
<tr>
<td></td>
<td></td>
<td>left and right, 107</td>
</tr>
<tr>
<td></td>
<td></td>
<td>link, 104</td>
</tr>
<tr>
<td></td>
<td></td>
<td>partitioned, 75</td>
</tr>
<tr>
<td></td>
<td></td>
<td>recursive, 110</td>
</tr>
</tbody>
</table>
Rudi de Louw is an SAP HANA architect for SAP South Africa, as well as an international speaker and trainer, principal consultant, and mentor. He has been working with SAP HANA for more than 5 years, and is the SAP HANA market unit champion. He is also the content owner for the SAP HANA professional certification examination. Rudi has provided software and technology solutions to businesses for more than 25 years, and has been with SAP since 1999. He has a passion for knowledge sharing and understanding, new technologies, and finding innovative ways to leverage these to help people in their development.

We hope you have enjoyed this reading sample. You may recommend or pass it on to others, but only in its entirety, including all pages. This reading sample and all its parts are protected by copyright law. All usage and exploitation rights are reserved by the author and the publisher.