DAU - Certified Data & Analytics Tester (CDAT)

Syllabus

Version 1.0

Released 06-08-2020

Copyright Notice

Thank you to the main authors

- Rogier Ammerlaan, Jaap de Roos and Armando Dörsek

Thank you to the review committee

Alexis Jesús Herrera Colmenares, Ángel Rayo Acevedo, Arjan Brands, Aurelio Gandarillas Cordero, Beatriz López Botello, Christine Green, Daniel Leo Lopez Romero, Egbert Bouman, Eibert Dijkgraaf, Emilie Potin-Suau, Fabiola Mero, Geoffrey Wemans, Hermon Alfaro Fuentes, Héctor Ruvalcaba, Jayapradeep Jiothis, Jean-Luc Cossi, Joan Gasull Jolis, Juan Pablo Rios Alvarez, Julie Gardiner, Kaan Sanli, Koos van Strien, Krishan Premcharan, Kyle Alexander Siemens, Márcia Araújo Coelho, Mario Alvarez Gómez, Melissa Pontes, Miaomiao Tang, Nadia Soledad Cavalleri, Neriman Kocaman, Petr Neugebauer, Dr. Ralph Elster, Ruth Margaret Florian Caipa, Santiago de Jesús Gonzalez Medellin, Samuel Ouko, Shreyansh Tewari, Dr. Srinivas Padmanabhuni, Tobi Steenbakkers, Thomas Cagley, Tim Koomen, Vanessa Islas Padilla, Wim Decoutere

Revision History

Version	Date	Remarks
0.2	June 8th, 2020	Initial BETA release
1.0	August 6th, 2020	After review of review committee

Table of contents

Business Outcomes

Business outcomes (BOs) are a brief statement of what you are expected to have learned after the training.

BO-1	Understand the complexity and elements in a Business Intelligence (BI) & Data Analytics (DA) environment
BO-2	Understand the specific aspects of testing in a BI & DA environment
BO-3	Have insights in the different skillsets required and the diversity of test roles/skills in a BI & DA environment
BO-4	Have insight in and can apply different test techniques and know how they can be practiced in a BI & DA environment
BO-5	Have insight in the different test aspects of a BI & DA environment compared to traditional testing, e.g. OLAP testing, Transformation testing, Completeness testing
BO-6	Understand the difference between quality attributes for system requirements and the specific quality attributes applied to data environments
BO-7	Understand and assess risk analysis and how to apply this in a BI & DA environment given the specific aspects of the environment
BO-8	Understand and assess the complexity of data and the importance of the use in DA
BO-9	Have knowledge about the complexity of DTAP environments (Development, Test, Acceptance and Production) in a BI & DA environment
BO-10	Understand the effects of privacy and data protection regulations for testing in DA projects

Learning Objectives/Cognitive Levels of Knowledge

To cover all Business Outcomes (BOs), each chapter Learning objectives (LOs) are defined. LOs are brief statements that describe what you are expected to know after studying each chapter, what can be examined for certification and are together with the BOs the traininggoals of this training

The LOs are defined based on Bloom's modified taxonomy **[BT1, BT2]** as follows:

Definitions	**K1 Remembering**	**K2 Understanding**	**K3 Applying**
Bloom´s definition	Exhibit memory of previously learned material by recalling facts, terms, basic concepts and answers.	Demonstrate understanding of facts and ideas by organizing, comparing, translating, interpreting, giving descriptions and stating main ideas.	Solve problems to new situations by applying acquired knowledge, facts, techniques and rules in a different way.
Verbs (examples)	Remember	Summarize	Implement
	Recall	Generalize	Execute
	Choose	Classify	Use
	Define	Compare	Apply
	Find	Contrast	Plan
	Match	Demonstrate	Select
	Relate	Interpret	
	Select	Rephrase	

Target Audience

This certification program is designed for test engineers, test coordinators, managers, business analysts, data leads, data warehouse developers and BI-consultants.

Prerequisites

Recommended

- Basic knowledge of testing in general, e.g. ISTQB or TMAP.
- Basic knowledge of Database- and Data Modeling principles, for instance BPMN or UML, ERD.

1 Introduction to Business Intelligence (BI) and Data & Analytics (DA)

In this introduction, terminology is defined about Business Intelligence, Data Warehouses, Data Analytics, Big Data, OLTP, OLAP, Data Marts, Reports and Dashboards.

Keywords

Business Intelligence, Data Warehouse, ETL, Data Marts, Reports, Dashboards, Analytics, Big Data, OLTP, OLAP, Three V's, Modelling, Audit Trail, Star Schema, Snowflake Schema, Slicing and Dicing, Facts, Dimensions.

LO-1.1	K1	Recall the definition of Business Intelligence
LO-1.2	K2	Understand challenges turning data into information
LO-1.3	K2	Explain the concept of a data warehouse, including types of data
LO-1.4	K1	Recall the functional view and the 'how view' of a data warehouse
LO-1.5	K1	Recall the aspects of the data mart
LO-1.6	K1	Recall the description of a data lake
LO-1.7	K2	Explain the description of data analytics
LO-1.8	K1	Recall the three main types of analytics
LO-1.9	K2	Explain the Big Data concept by 3 V's
LO-1.10	K2	Recall additional V's associated with Big Data
LO-1.11	K1	Recall the definition of Data Mining
LO-1-12	K2	Explain and understand the difference between OLTP and OLAP
LO-1.13	K2	Explain cubes, facts, slicing and dicing
LO-1.14	K2	Understand what reports and dashboards are meant for
LO-1.15	K2	Recall the different modelling techniques associated with data warehouse design
LO-1.16	K2	Explain the ETL process and Source to Target mappings
LO-1.17	K2	Understand Localization and Internationalization
LO-1.18	K2	Understand the impact of (different) Character Sets on the BI & DA environment

In the Business Intelligence & Data Analytics (BI & DA) environment different terms are being used for different parts of the model shown in Illustration 1. In this chapter a common understanding of what we see as the world of BI & DA is described. This doesn't mean other definitions don't exist, but for the content of this syllabus it is relevant that everybody gets the same understanding of the definitions used.

This chapter describes different aspects of a BI & DA environment. Illustration 1 will be used to explain the different parts of the model. It is important for a tester to understand the terminology and the environment.

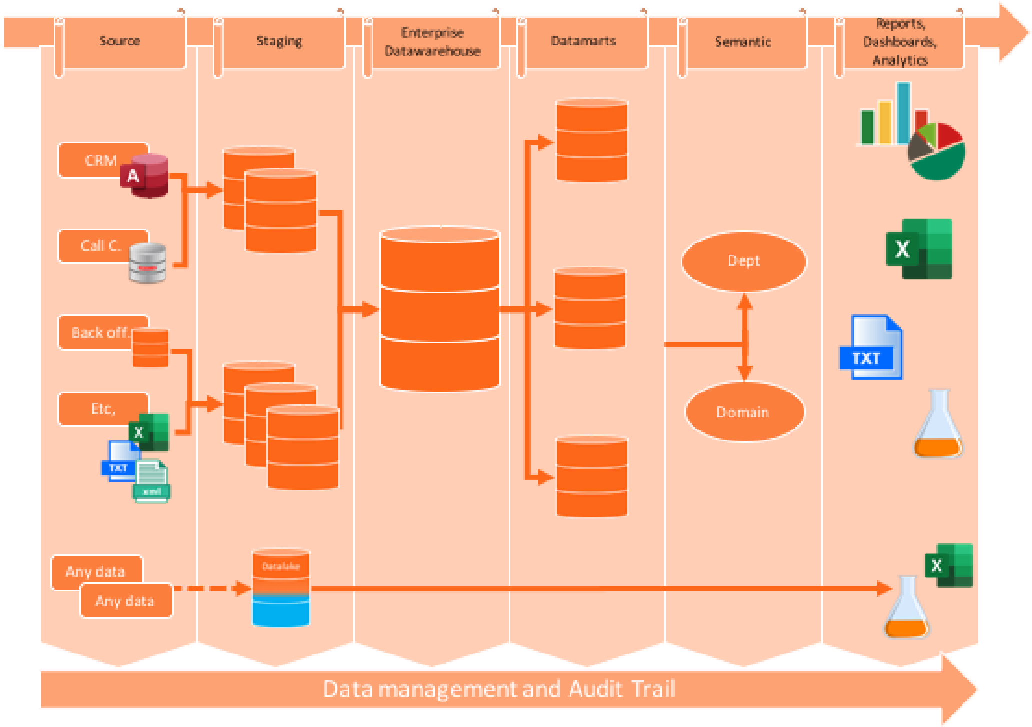


Illustration 1 A typical BI environment

1.1 Business Intelligence (LO-1.1)

Business Intelligence (BI) comprises the strategy, methods and technologies used to gather, store, report and analyze business data to help people analyze data to make business decisions.

BI is further referred to as the way to visualize this analysis in a meaningful and clear representation. BI therefore refers to the "front end" technology in the DA environment. In the wider context, the concept of BI relates to the process of generating "knowledge" from data, including concepts like data warehousing. BI technologies like Tableau, PowerBI, SAS, can handle large amounts of structured and sometimes unstructured data to help report the current state, but also to identify, develop, and otherwise create new strategic business opportunities.

1.2 Challenges when turning corporate data into information (LO-1.2)

The data which is needed to turn into meaningful information may come from several sources. Integrating these data to start analyzing may lead to challenges. The source systems were often built for different purposes than informing CEO's or feeding data scientists with useful data – in the right format and at the right moment.

For instance, the source systems may be running on premise of our company or somewhere 'in the cloud'. Data may be delivered in batches, through pull systems like database queries or through push systems like messaging systems. Data may be received periodically (e.g. per batch run, daily or weekly) or continuously and in real time, like with feeds from energy meters. Some may store historical data; others may store only the actual values. It can already be the owners of the data – or need to buy this data from others (Social Media, Weather reports, Traffic information). The format may be proprietary or open, data quality may be taken care of by its owner or seen as a problem of data analysts.

To counter those differences and to make sure that there is no overload of the source systems with defined queries, a collection of data is stored in a different place so it can be used at the moment that is convenient. By adding timestamps to the data, historic information is introduced and enable analysis about the past periods and spot trends.

1.3 Data Warehouse (LO-1.3, LO-1.4)

A Data Warehouse enables better BI & DA. Data warehouses are designed to overcome most of the challenges mentioned in the previous chapter.

In the center of the model of Illustration 1 you find the data warehouse. A data warehouse, also known as Enterprise Data Warehouse (EDW), stores data from several sources, either direct or via a staging area (is an intermediate storage area used for data processing during the extract, transform and load process). It is a consistent way of storing (company) data, both current and historical, mainly associated with structured data. It consists of a data model regardless of the original source or staging area. The performance is optimized for processes storing new data. The data model of the data warehouse is created via different data modelling techniques, like: re-structuring data into Facts and Dimensions and de-normalization.

Definitions of a Data Warehouse
To get insight of a Datawarehouse we present the (different) definitions of two important authors on the subject of BI and Data Warehousing, namely W.H. (Bill) Inmon and R. (Ralph) Kimball.

W.H. (Bill) Inmon: “A data warehouse refers to a subject oriented, integrated, time-variant, non-volatile collection of data in support of management’s decision-making process”.

Subject Oriented
Creating a “subject oriented” collection of data results in the situation where all data regarding a subject, for instance “customer” or “product” is brought together, independent of its source.

Integrated
In an integrated collection of data, the same name is used for the same information (e.g. “customer-number” vs “client-id”), coding is harmonized (e.g. “male/female” vs “man/woman”), the same measurements are used (e.g. USD vs EUR and inches vs centimeters).

Time Variant and Non-Volatile

In operational systems data is continuously added (inserted), edited (updated) and deleted, providing an "actual" view of the situation. A data warehouse is used to provide snapshots of the situation at any certain moment in time and *over* time. Therefore, timestamps are required, introducing "history" (also known as: slowly changing dimensions (SCD)). Data is not physically deleted but only "flagged" as such. Changes are added not by updating existing records but by adding new records.

This definition seems to stress the *writing* part of the data warehouse environment, as it describes data modeling choices that enable fast and easy writing (inserts, updates, deletes) of data into the data warehouse.

Ralph Kimball refers to a Data Warehouse as "a copy of transaction data specifically structured for query and analysis". This definition fits the "Business Intelligence" side very well, as it stresses the need of easy and fast querying (writing and reading) of the data that is made available in Data Marts (see chapter 1.4). In chapter 1.11 more information on data modelling aspects can be found. A transaction, in this definition, is a set of logically related operations. For example, you are transferring money from one bank account to another.

1.4 Data Mart (LO-1.5)

Data Mart (DM), mentioned in Illustration 1, is a subset of the Enterprise Data Warehouse (EDW). The DM will hold a subset of the data in the EDW, focusing on a single subject area and it is oriented towards a specific task or department/user group, enabling them to perform queries fast and in an easy manner. It presents data in a way that the people of the department will understand. It also holds summarized data, whereas the data warehouse will hold all details.

It is designed as a Dimensional Model ("star schema", "snowflake") and focuses on integration of information regarding a certain (single) subject area. Data Marts can also tackle issues regarding privacy and security, for instance by showing less detailed information about individuals – or only to authorized colleagues.

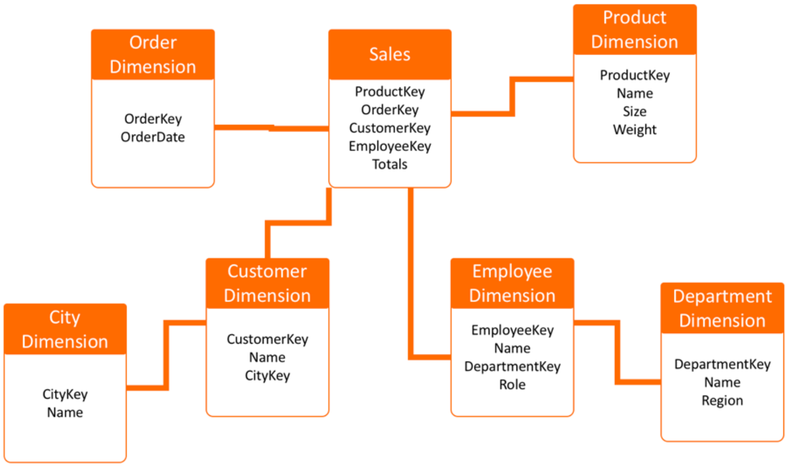


Illustration 2 Example Snow Flake Model

1.5 Data Lake (LO-1.6)

A data lake is a large repository (Big Data) of internal and external data in its natural, unstructured form as a raw data reservoir. In Illustration 1 it is illustrated on the left-hand bottom end. Raw data flows into a data lake in real time, and users can segregate, correlate and analyze different parts of data based on their needs or as a source for the ETL (Extract, Transfer, Load) process of a Data Warehouse.

In a Data Warehouse, data is often stored in tables (rows and columns). This requires that attributes of the data (e.g. data types, length) must be known upfront. Data lakes store (semi-structured) data types in their native format, without requiring the tables to be defined upfront. A data lake can include structured data from relational databases (rows and columns), semi-structured data (CSV, logs, XML, JSON), unstructured data (emails, documents, PDFs) and binary data (images, audio, video). Information should be tagged with a set of extended meta data tags (Campbell**[CC1])**.

Metadata is a description of a digital asset, 'data about data'. Terms associated with an object to describe it, for instance the date a record was created or added, the source where the information comes from, the type or category of the data. Metadata helps to organize digital assets and make it therefore easier to find those specific assets within a group of objects.

Data lakes rely on low cost storage options because of the volume of its contents. The infrastructure will need to fit the type of data stored in the data lake: large volumes of unstructured data. The core technology was based on the Apache Hadoop Ecosystem, an open source software framework that distributes data storage and processing among commodity hardware located in on-premises data centers. Hadoop includes a file system called HDFS that enables customers to store data in its native form.

As the cloud computing industry matured, object stores from Amazon, Microsoft, and other vendors introduced interim data lake solutions, such as Amazon Simple Storage Service (S3), Microsoft Azure Blob, and Google Cloud Storage.

Relational databases are often queried by use of SQL, Standard Query Language. As the data lakes are often stored in non-relational database systems or so called NoSQL (“Not only SQL”) databases, testers are well advised to master skills associated with querying data in these specific systems, too.

1.6 Data Analytics (LO-1.7, LO-1.8, LO-1.10)

Data analytics, which mainly take place in the phase Reports, Dashboards, Analysis of illustration Illustration 1, comprises the combined corporate and external data sources for the creation of analytical models. Based on these models, decisions can be made regarding groups and individual subjects. Data Analytics is strongly associated with ‘Big Data’.

A popular definition of analytics is "the extensive use of data, statistical and quantitative analysis, explanatory and predictive models, and fact-based management to drive decisions and actions" **[DA1].**

There are three types of analytics: Descriptive Analytics, Predictive Analytics and Prescriptive Analytics (Davenport**[TD1]**).

1.6.1 Descriptive Analytics

Reports, dashboards, alerts and scorecards are used to describe what has happened in the past. Descriptive Analytics may also be used to classify customers or other entities into groups that are similar on certain dimensions.

1.6.2 Predictive Analytics

With Predictive Analytics, using data from the past to create models the future is predicted. Todays most used technique for creating these predictive models is machine learning. Typically, multiple variables are used to predict a particular *dependent* variable. For instance, using credit history of clients to predict the ability to repay their loans in the future.

1.6.3 Prescriptive Analytics

Predictive analytics uses a variety of quantitative techniques (such as optimization) and technologies (e.g., models, machine learning and recommendation engines) to specify the optimal behaviors and actions (Davenport**[DA1]**). It will tell you what to do, based on measuring cause and effect in test groups and control groups. E.g. a drug company tests his product on a group of test subjects and gives a control group a placebo to verify the effect. If a statistically significant difference is discovered between the two, it will tell you which approach to take.

In prescriptive analytics data can have a correlation with each other, for instance where two or more things (or data elements) have a mutual relationship or connection, or causation, that is the influence of causes on effects in a later state of the data.

Correlation vs Causation
Note that while causation and correlation can exist at the same time, correlation does not imply causation. Causation explicitly applies to cases where action A causes outcome B. On the other hand, correlation is simply a relationship between A and B. Action A relates to Action B—but one event doesn't necessarily cause the other event to happen.

1.7 Big Data (LO-1.9)

Big data is a combination of structured, semi-structured and unstructured large amount of data collected by organizations that can be analyzed for predictive modeling and other advanced analytics applications. This collection mainly take place in the phases Source, Staging and EDW of illustration 1, In a 2001 research report, META Group (now: Gartner) analyst Doug Laney characterized big data by using three V's, i.e. increasing volume (amount of data), velocity (speed of data in and out), and variety (range of data types and sources). Although big data doesn't equate to any specific volume of data, big data deployments often involve terabytes (TB), petabytes (PB) and even exabytes (EB) of data captured over time.

1.7.1 Volume, Velocity, Variety (LO-1.9)

Volume
Volume refers to the amount of data is created, stored, analyzed and visualized. The sheer volume of the data is enormous, and a very large contributor to the ever-expanding digital universe is the Internet of Things (IoT) with sensors all over the world in all devices creating data every second. And don't forget all Social Media or Market-basket analysis off course.

Velocity
Velocity refers to the speed at which the data is created, stored, analyzed and visualized. In the past, when batch processing was common practice, it was normal to receive an update from the database every night or even every week. Nowadays Real Time Data, streaming data, sensing data is more and more used. Also because of IoT this becomes enormous important.

Variety
Variety refers to the many different formats of data. In the past, all data that was created was structured data: it fitted neatly in columns and rows and in text files. Today, most of the data that is generated is unstructured data. Data today comes in many different formats: structured data, semi-structured data, unstructured data and even complex structured data. The wide variety of data requires a different

approach as well as different techniques to store all raw data – and to analyze and to effectively use the data.

Companies use the big data accumulated in their systems to improve operations, provide better customer service, create personalized marketing campaigns based on specific customer preferences and, ultimately, increase profitability.

1.7.2 Additional V's (LO-1.10)

In the course of time, additional V's were added to the big data definition, explaining important aspects of big data and a big data strategy that organizations cannot ignore. Four of these are: Veracity, Variability, Visualization and Value (Rijmenam**[DF1]**).

Veracity
Organizations need to ensure that the data is *correct* as well as the analysis performed on the data are correct. Especially in automated decision-making, where no human is involved anymore, you need to be sure that both the data and the analysis is correct to tell the through in what you present.

Variability
Big data is extremely variable, i.e. the meaning of the data may differ from time to time, just like two words may have a different meaning. This will depend on the context and will be important in processes like sentiment analysis. Variability is often confused with variety, but where variety is concerned with the way data is stored and presented, variability is concerned with the actual meaning of the data.

Visualization
With the right analyses and visualizations, raw data can be put to use, otherwise will remain essentially useless. Though perhaps not the most challenging part from a technical perspective, telling a complex story in a graph is very difficult but also extremely crucial to present it in such a way that the recipients understand the message of the story.

Value
More and more organizations understand that data must be perceived as an important asset. Of course, data itself is not valuable at all. The value is in the analyses done on that data and how the data is turned into information and eventually turned into knowledge, which could be used for learning objectives or just to create more knowledge of a certain subject. The value is in how organizations will use that data and turn their organization into an information-centric company that relies on insights derived from data analyses for their decision-making.

1.8 Data Mining (LO-1.11)

Data mining concerns the process of discovering new patterns from large data sets involving methods at the intersection of artificial intelligence, machine learning, statistics and database systems. This implies a discovery of trends and patterns in data – not by humans, but by the computer itself. (Davenport**[TD1]).** Data mining is most commonly associated with statistical and predictive models (see: Predictive Analytics). Data mining techniques use algorithms. Data mining mainly take place in the phase Reports, Dashboards, Analysis of illustration 1,

Some examples of data mining applications are:

- detection of fraud,
- predicting stock market price,
- analysis of customer behavior (e.g. in "Market Basket Analysis").

1.9 OLTP vs OLAP (LO-1.12, LO-1.13)

Several differences are distinguished between the generic operational systems and the environments used in data analytics.

Operational systems often also called On Line Transaction Processing (OLTP), is used for support of primary business activities like processing customer requests, invoice handling, processing lab results, planning courses and filling or filing tax forms (mainly 'writing' and 'updating' data). Whereas On Line Analytical Processing (OLAP) is used to create value from the existing business data, by judging measures like company results over several dimensions like Time, Pricing Categories, Regions and Customer Groups (mainly 'reading' data).

In the table below you can find some of the differences between On Line Transaction Processing (OLTP) and On Line Analytical Processing (OLAP).

	OLTP System Online Transaction Processing (Operational System)	OLAP System Online Analytical Processing (Data Warehouse)
Source of data	Operational data; OLTP's are the original source of the data	Consolidation data; OLAP data comes from the various OLTP sources
Purpose of data	To control and run fundamental business tasks	To help with planning, problem solving, and decision support
Data insights	Reveals a snapshot of ongoing business processes, actual status	Multi-dimensional views of various kinds of business activities, time series
Inserts, Updates and Deletes	Swift inserts, updates and deletes initiated by end users	Periodic long-running batch jobs refresh the data, no physical deletion of data (but flagging "deleted" records)

	OLTP System Online Transaction Processing (Operational System)	OLAP System Online Analytical Processing (Data Warehouse)
Queries	Relatively standardized and simple queries returning relatively few records	Often complex queries involving joins, transformations and aggregations
Processing Speed	Typically, very fast processing, focused on real time informing of end users and typically short batch windows	Depending on the amount of data involved and the complexity of the (batch) processing, loading data and querying for results may take many hours
Space Requirements	Can be relatively small if historical data is absent or archived	Larger due to the existence of redundant storage, aggregation structures and history data
Database Design	Highly normalized, with many tables	Typically, de-normalized with fewer tables, e.g. use of star- and/or snowflake schemas for easy reporting
Backup and Recovery	As operational data is mostly critical to run the daily business, backups and restores are of the utmost importance	Instead of creating backups (and restores) re-loading the OLTP-data may be sufficient as a recovery method, dependent on processing times and importance of certain analytics products

Table 1 Differences OLTP vs. OLAP

1.9.1 OLAP Cubes and Dimensional Reporting

Reports and dashboards are great for visualizations that present data in a uniform fashion. But when analytical users need to be able to explore data in an interactive way, clicking through regions, departments, turnover by month etc., the OLAP cubes come into play.

An OLAP cube, also known as multidimensional cube or hypercube, is a data structure which offers ways to display and sum large amounts of data and provide searchable access to any data points. The data can be rolled up, sliced, and diced as needed to handle all kind of questions that arise when confronted with the data set. In contrary to most OLTP databases, which are not designed for analysis, OLAP databases are specialized databases that are designed to help extract this business intelligence information from the data.

In order to enable easy and fast querying, the underlying data model is often based on Kimball data modeling techniques. The star schemas and snowflake schemas (as opposed to Relational Modeling techniques) will hold Fact and Dimension tables which are used in the cube. See §1.12: Data Modeling.

Due to its structure, the OLAP cube often accessed by other means than SQL by multidimensional expressions (MDX[1]). This because SQL is not sufficient enough with its technique, where records are inserted one by one, is extremely slow for the vast volume of data to be loaded in the DWH. Several

[1] Multidimensional Expressions (MDX) is a query language for online analytical processing (OLAP) using a database management system. Much like SQL, it is a query language for OLAP cubes. It is also a calculation language, with syntax similar to spreadsheet formulas

analytical tool vendors like Tableau, Cognos, Microsoft PowerBI and SAS provide software to perform OLAP.

1.9.2 Key aspects in OLAP systems

Measures

Measures are the numeric values that users want to slice, dice, aggregate, and analyze; they are one of the fundamental reasons why you would want to build OLAP cubes using data warehousing infrastructure. For instance, Gross- and Nett Earnings, Direct- and Indirect Costs, number of products sold.

Dimensions

Dimensions allow the filtering, grouping, and labeling of data. For instance, Regions or Locations, Dates and Product Dimensions. The data can be presented in a format where the data is categorized naturally into these hierarches and categories to allow a more in-depth analysis. Dimensions may also have natural hierarches to allow users to "drill down" to more detailed levels of detail. For instance, the date dimension that can be drilled down from Year to Quarter, Month, Week and Day.

Drill Down and Drill Through

"Drilling down" into the data in an OLAP cube, means that the user is analyzing the data at a different level of summarization. The level of detail of the data changes as the user drills down. Here terms like granularity (level of detail) are important, as the user is moving from summarized data to data with a narrower focus.

When users "drill through" data, they are requesting the individual transactions that contributed to the OLAP cube's aggregated data at a lowest level of detail for a given measure value.

So, the difference between them is that a drill-down operates on a *predefined hierarchy* of data, for instance from Europe, to The Netherlands, to Amsterdam within the OLAP cube. A drill-through goes directly to the lowest level of detail of data and will retrieve a set of rows from the data source that has been aggregated into a single cell.

Granularity

The most detailed unit of the data is a fact, a contract, invoice, spending, task, etc. Each fact might have a measure – an attribute that can be measured, such as: price, amount, revenue, duration, tax, discount, etc. The "grain" (or granularity) of a fact table states the level of detail in the fact table. A low level of granularity means that the table holds very detailed information – which can later be "rolled up".

1.10 Reporting and Data Visualization (LO-1.14)

Visualizing data using charts, graphs, and maps is one of the most impactful ways to communicate complex data. Visualizing data for end users, is a very important aspect of the data analytics activities and require strong analytical and communication skills next to knowledge on the business and (financial) reporting.

There are more standards regarding reporting, for instance, those of the International Business Communication Standards (ICBS). They feature important aspects like storylines (Barbara Minto), Perceptual Rules (Few**[SF]**), and Business Rules (Hichert**[RH]**).

Report

Report according to Poonam Madan is defined as "a document that presents information in an organized format for a specific audience and purpose" (Madan**[PM1]**). There are two types of reports: static and dynamic reports. Static reports are run immediately upon request, and then stored with its data and will not change anymore. Dynamic reports are created at runtime. Each time a dynamic report is run, it gathers the most recent data in the Data Warehouse. Only the report definition, which remains the same over time, is stored.

Dashboard

A dashboard is a visual display of the most important information needed to achieve one or more objectives; consolidated and arranged on a single screen so the information can be monitored at a glance **[SF1]**.

Dashboards vs Reports

A report is a more detailed collection of tables, charts, graphs and text and it is used for a much more detailed, full analysis while a dashboard is used for monitoring what is going on. The behavior of the pieces that make up dashboards and reports are similar, but their makeup itself is different. A dashboard answers a question in a single view and a report provides information. Put in another way, the report can provide a more detailed view of the information that is presented on a dashboard.

1.11 Data Modeling (LO-1.15)

In order to store data in the environment in an orderly fashion, data modeling techniques will be used. Engineers and testers need to understand reasons for applying certain techniques – and their effects.

In "regular" computer systems (see: OLTP) data is usually stored in a normalized fashion. Edgar Codd introduced rules for data modeling: Normal Forms. Normalization divides larger tables into smaller tables and links them using relationships. The purpose of normalization is to eliminate redundant data and to ensure that data is stored logically. Designing database tables according to these rules lead to a highly normalized logical data model (LDM).

A drawback of highly normalized data tables is that querying the data, through linking tables based on relationships between them ("foreign keys"), can be hard, time consuming and error prone. Therefore, Data Marts usually are designed as in a de-normalized[2] fashion, leading to data models that are designed according to other design rules.

Well known designs are *star schemas* or *snowflake* **[GUR]** which are multi-dimensional schemas, folding Fact- and Dimension tables that are designed for the analytical purpose (OLAP). Inside the Data Warehouse design techniques are found, like (Business/Raw) Data Vault (Linstedt**[DL1]**) and Anchor Based Modeling (Rönnback**[LR1]**).

1.11.1 Maintaining History: Slowly Changing Dimensions

Earlier, the feature of a data warehouse is that most changes in the data are recorded by "adding history", not by deleting or editing the actual records themselves. This introduces the Slowly Changing Dimensions or SCD types. The following SCD types are commonly used in data warehouse design.

Type 0 – No Changes
If SCD Type 0 is applied, dimensions never change. Changes are not allowed at all.

Type 1 – No History
With SCD Type 1, existing values are simply overwritten with the new data. The effect is that tables do not grow much, because no rows are added to handle the time aspects. The drawback of this is that the table will always only hold the current value for each attribute and any historical value of the data is lost.

Type 2 – Row Versioning
When using SCD Type 2 new record are added, encompassing the change and flag the old record as inactive (by using an "inactive" column and/or a start date and end date). This allows the fact table to continue to use the old version of the data for historical reporting purposes and use the changed data in the new record to only impact the fact data from that point forward.

Type 3 – Previous Value column
Using SCD Type 3, you add an extra column to store the most recent past value of the column(s) you wish to be able to report on. When the data is updated the existing value is "moved" to the column defined to store the previous past value and the new value is placed into the reportable column. This allows you the ability to look back at what the value of the data was previously. Note that loading/updating data may become very difficult and error prone.

[2] Denormalization is a strategy used on a previously normalized database to increase performance. In computing, denormalization is the process of trying to improve the read performance of a database, at the expense of losing some write performance, by adding redundant copies of data or by grouping data.

Type 4 – History Table

With SCD Type 4, current values are stored in the dimension table but track all changes in a separate table. This provides an easy way to show the current data, selecting historic data may become more challenging. One of the reasons to introduce this SCD-type can be database performance: this way the database table with “current” data can be kept small (less records).

1.12 The ETL-Process (LO-1.16)

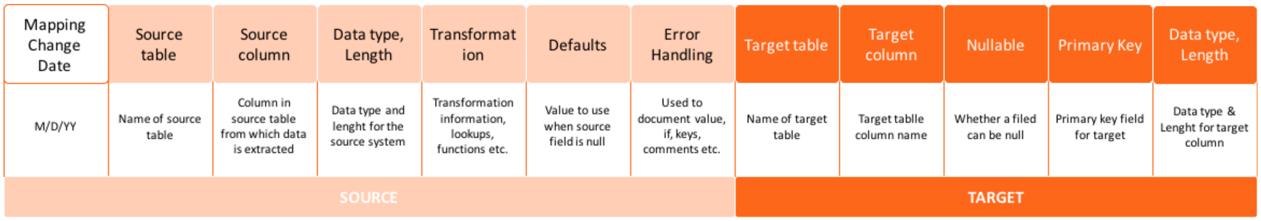
ETL stands for: Extract, Transform and Load (data). Data is *extracted* from the source systems, *transformed* in a usable format and *loaded* into the (enterprise) data warehouse.

ETL is commonly associated with Data Warehousing projects, but, in reality, any form of bulk data movement from a source to a target can be considered ETL. ETL processes are seen, when the need arises to move bulk application data from one source to another for data integration or data migration purposes. ETL testing is a data centric testing process to validate that the data has been transformed and loaded into the target as expected **[DG1]**.

Testing the ETL process may cover aspects like completeness, auditability, logging processing results, exit codes, processing times, re-starting of ETL processes, syntactical and semantical validations (are the right source fields moved to the right target fields and do they indeed mean the same, were the right tables joined) and are the conversions of dates, numbers and strings performed as expected.

ETL is documented by means of Source-to-Target Mappings (STM) (a.k.a: ETL mapping documents), which will often describe:

- Names of source and target tables,
- Names (and data types) of columns in the source- and target tables,
- How to detect Inserts, Updates and Deletes,
- The way SCD are applied (adding History),
- Transformations of data.



Mapping Change Date	Source table	Source column	Data type, Length	Transformation	Defaults	Error Handling	Target table	Target column	Nullable	Primary Key	Data type, Length
M/D/YY	Name of source table	Column in source table from which data is extracted	Data type and lenght for the source system	Transformation information, lookups, functions etc.	Value to use when source field is null	Used to document value, if, keys, comments etc.	Name of target table	Target tablle column name	Whether a filed can be null	Primary key field for target	Data type & Lenght for target column
SOURCE							TARGET				

Illustration 3 Source to Target Mapping Header Example

The ETL process can be a very compute-intense activity. Due to a shift to data lakes and cloud-based data warehousing solutions, in which the complete set of data is first moved to the cloud (storage) and then transformed, the abbreviation ELT is being introduced: Extract, Load and Transform. This stresses that the activity of Transformation is being performed where it makes most sense, i.e. not on premise but "in the cloud".

1.13 Internationalization and Localization (LO-1.17)

There is a difference in adaptation of for instance a product, application or document when it will be created for a Local or International market. In the next paragraphs the differences will be explained.

1.13.1 Localization

According to the W3C internationalization site, localization refers to the adaptation of a product, application or document content to meet the language, cultural and other requirements of a specific target market (a local).

Amongst others, it can entail customization related to:

- Numeric, date and time formats,
- Use of currency,
- Keyboard usage,
- Collation[3] and sorting,
- Symbols, icons and colors.

1.13.2 Internationalization

Internationalization is the design and development of a product, application or document content that enables easy (future) localization for target audiences that vary in culture, region, or language. Internationalization typically entails:

- Designing and developing in a way that removes barriers to localization or international deployment. This includes such things as enabling the use of proper encoding of characters.
- Providing support for features that may not be used until localization occurs. For example, using markup in your DTD[4] (Document Type Definition) to support bidirectional text[5], or for identifying language.
- Enabling code to support local, regional, language, or culturally related preferences. Examples include date and time formats, local calendars, number formats and numeral systems, sorting and presentation of lists, handling of personal names and forms of address, etc.
- Separating localizable elements from source code or content, such that localized alternatives can be loaded or selected based on the user's international preferences as needed.

[3] Collation specifies how data is sorted and compared in a database

[4] A DTD defines the structure and the legal elements and attributes of an XML document.

[5] A bidirectional text contains two text directionalities, right-to-left (RTL or dextrosinistral) and left-to-right (LTR or sinistrodextral). It generally involves text containing different types of alphabets, but may also refer to boustrophedon, which is changing text direction in each row.

1.13.3 Importance of Localization and Internationalization

Internationalization and Localization are important to Data Analytics engineers and testers. When collecting data from source systems, the way data is stored needs to be clear. Each interface to the data (both upon input as well as when retrieving the data) can influence the way the data is stored and/or presented. At processing the ingested data[6], handling internationalization and localization concepts is important, not only from a pure technical perspective, but also when looking at business rules: should customer “AEL Dörsek” be treated as a copy of “AEL Doersek” and “AEL D¿rsek”? Or are these to be treated as individual customers? This touches the subject of data quality (see chapter 5).

When presenting data, correct implementation of localization is important: tools should be able to handle differences in currencies, thousands-separators, decimal separators, etc. Are characters being displayed correctly; and the non-Latin characters like Chinese, Japanese, Arabic, Greek, too?

1.13.4 Character Sets and Internationalization (LO-1.18)

In the course of history, different character sets were designed to store data. Below, is described a few of these character sets including their main features.

1.13.4.1 ASCII

ASCII is a 7-bit encoding technique which assigns a number to each of the 128 characters that are most frequently used in American English. This allows most computers to record and display basic text. It does not include symbols frequently used in other countries, like the British Pound sign or characters with the German Umlaut. ASCII is understood by almost all email and communications software.

1.13.4.2 ISO/IEC 8859 (Extended ASCII)

ISO/IEC 8859 is an 8-bit extension to ASCII developed by ISO (the international organization for standardization). Next to the 128 ASCII characters it covers characters like the Euro sign (#128), British Pound (#163) and the American Dollar Cent (#162) symbol. Several variations of the ISO/IEC 8859/Latin standard exist, to cover different language families. For instance, Latin-1 for Western European languages, Latin-5 for Turkish, ISO/IEC 8859-5 for Cyrillic and ISO/IEC 8859-8 for Hebrew. Depending on the chosen standard, the stored bytes will present (partially) different characters.

[6] Data ingestion is a process by which data is moved from one or more sources to a destination where it can be stored and further analyzed. The data might be in different formats and come from various sources, including RDBMS, other types of databases, S3 buckets, CSVs, or from streams

1.13.4.3 UNICODE

Unicode is an attempt by ISO and the Unicode Consortium to develop a coding system for electronic text that includes every written alphabet in existence. It uses 8, 16 and 32-bit characters (=multi byte characters) depending on the specific representation. This results in larger documents because storing these characters take up more space than ASCII or Latin-1 based texts. Note: The first 256 characters of UNICODE are identical to Latin-1. Unicode will even store emoji's (e.g. smileys) and most mathematical symbols.

1.13.4.4 Possible effects of mixing character sets

Amongst others, the following issues may occur when mixing character sets:

- Possible ambiguity in presentation and conversions as bytes may be presented in the wrong character set,
- Storage may not be sufficient when storing multi byte characters[7] is sometimes not foreseen at the design stage,
- When poorly implemented, the sort order of the bytes can be ambiguous: which byte of Multi Byte Character should be read first and which one last (big endian, little endian)[8],
- When using string functions, programming languages and database systems may have difficulty in selecting the correct character (selecting the second byte is not the same as selecting the second character),
- Storing special characters, like emoticons, may be supported by specific applications but not by the tools in which reports and dashboards are developed,
- characters that appear the same, might not be recognized as such. (e =/= e in different character sets for example).

[7] A multi-byte character is a character composed of sequences of one or more bytes. Each byte sequence represents a single character in the extended character set.

[8] A big-endian system stores the most significant byte of a word at the smallest memory address and the least significant byte at the largest. A little-endian system, in contrast, stores the least-significant byte at the smallest address. Computers store information in various sized groups of binary bits.

2 Data & Analytics Testing Strategy

In this chapter are the characteristics of testing explained, different test methods, risk analysis and discuss differences and similarities of testing in a data & Analytics environment and skills of a tester.

Keywords

Testing, BI Testing, Test process, Waterfall, V-model, Scrum, Agile Testing, Risk Analysis, Test leader, (Reporting, ETL, Data Migration or Data Quality) Tester

LO-2.1	K1	Recall the definition of Testing
LO-2.2	K1	Recall the Test methodologies
LO-2.3	K2	Understand the different test processes
LO-2.4	K2	Explain BI testing
LO-2.5	K2	Explain Scrum and Agile Testing
LO-2.6	K1	Recall the Agile BI Checklist and BI Best Practices
LO-2.7	K2	Explain Risk based testing
LO-2.8	K3	Recall and apply a risk-based testing strategy
LO-2.9	K2	Explain the definition of Risk identification
LO-2.10	K2	Explain how likelihood and impact Influence Risk
LO-2.11	K1	Recall the definition of Test Leader and Tester
LO-2.12	K1	Recall the Tester's skills and additional skills for D&A testers
LO-2.13	K2	Explain the differences of skills for the Reporting Tester, ETL Tester, Data Migration Tester and Data Quality Tester
LO-2.14	K3	Recall and set-up a test strategy and approach in a BI environment

2.1 Testing and Test Methodologies (LO-2.1, LO-2.2, LO-2.3)

There are different definitions of testing, one from the ISTQB syllabus:

*"The process consisting of all lifecycle activities, both static and dynamic, concerned with planning, preparation and evaluation of software products and related work products to determine that they satisfy specified requirements, to demonstrate that they are fit for purpose and to detect defects. " (ISTQB**[IS1]**)*

A common perception of testing is that it only consists of running tests, i.e., executing the software (mainly called dynamic testing). This is part of testing, but not all of the testing activities. There are also test activities before and after test execution. There are several test methodologies that define a structured test process. These activities include planning, monitoring and control, choosing test conditions, designing and executing test cases, checking results, evaluating entry- and exit criteria, reporting on the testing process and system under test, and finalizing or completing closure activities after a test phase has been completed. Testing also includes reviewing documents (and/or source code) and conducting static analysis, so called as part of static testing.

Both dynamic testing and static testing can be used as a means for achieving similar objectives and will provide information that can be used to improve both the system being tested and the development and testing processes.

Testing, if it is in for instance an app environment or a BI & DA environment, can have one or even more of the following objectives:

- Finding defects
- Gaining confidence about the level of quality
- Providing information for decision-making
- Preventing defects

The thought process and activities involved in designing tests early in the life cycle (verifying the test basis via test design) can help to prevent defects from being introduced into code. Reviews of documents (e.g. requirements, use cases) and the identification and resolution of issues also help to prevent defects appearing in the code.

Different viewpoints in testing take different objectives into account. For example, in development testing (e.g., component, integration and system testing), the main objective may be to detect as many failures as possible so that defects in the software are identified and can be fixed. In acceptance testing, the main objective may be to confirm that the system works as expected, to gain confidence that it has met the requirements. In other cases, the main objective of testing may be to assess the quality of the software (with no intention of fixing defects), to give information to stakeholders of the risk of releasing the system at a given time. Maintenance or regression testing often includes testing that no new defects have been introduced during development of the changes. During operational testing, the main objective may be to assess system characteristics such as reliability or availability.

Different approaches are developed over the decades, but most often used testing methodologies are:

- ISTQB: Internationally most common and known. A test approach on how to setup a structured test process in different software development life cycles (agile, scrum, waterfall or V-model approaches)
- TMap®, a test management approach on how to setup a structured test process in different software development life cycles (agile, scrum, waterfall or V-model approaches)
- Quality for DevOps Teams, a test approach based on the Knowledge of Body Tmap® about testing in a DevOps environment

But different countries and organizations could also develop their own approaches based on combinations of the above-mentioned methods, and the best practices of their environment.

2.2 BI & DA Testing in a traditional approach (LO-2.4)

BI & DA testing holds several aspects in itself that are quite different from a regular software test project. But knowledge of test methodologies is still important for understanding, planning and organizing a test project.

For a definition of testing a reference is made to (ISTQB**[IS2]**), but specifically in the context of a Data, Analytics or Business Intelligence environment an addition on the traditional test definitions would be in its place.

"Business Intelligence & Data Analytics Testing is the process of validating the data, format and performance of the ETL, analytics, reports, subject areas and security aspects of the BI & DA projects."

Main focus areas:

- Data at the source
- Data transformation
- Data loading
- Reporting

Because of these specific aspects of BI testing, it is important for a tester that these aspects are given a place in the software development lifecycle model on the different test levels, as addition on the applicative changes on a product. For instance, data is an important part of the 'system' under test and therefore, testing of data (and possibly data quality) is part of functional testing. Integration testing can apply to the flow of data from source to target system. All elements of the BI & DA environment that are in scope of the project should be mapped on the several phases of the development life cycle.

For instance, data transformation could be part of the component test, where the development team validates the individual components developed for the data transformation, whereas a possible separate test team will validate the functional business rules of data transformation on system test level.

For each aspect of the BI test a test approach needs to be chosen on the several test levels. In practice, there could be more, fewer or different levels of development and testing, depending on the BI environment.

Software work products (such as data transformation documents, business rules but also business scenarios or user stories, use cases, requirements specifications, design documents and code) produced during development are often the basis of testing. Therefor implementing reviews of these documents can help to improve the quality of the documents and improve the quality. So, don't focus only on the execution of testing with for instance running transformations, but also on improving the quality of the description of the rules, to prevent wrong interpretations.

2.3 BI & DA Testing in an agile approach (LO-2.5, LO-2.6)

Within Agile many agile frameworks are developed. Scrum is one of the agile frameworks that is most commonly used. Scrum is founded on empirical process control theory, or empiricism. Empiricism asserts that knowledge comes from experience and making decisions based on what is known. Scrum employs an iterative, incremental approach to optimize predictability and control risk. Three pillars uphold every implementation of empirical process control: transparency, inspection, and adaptation (Schwaber**[SG1]**).

Agile Testing is an iterative test approach based within the context of the Agile way of working. There are many definitions of Agile Testing (C. Kaner, M. Bolton, J. Bach and many more), but a good explanation is mentioned in chapter 1.4 of the Agile United Syllabus. Interesting to read are also the books ´Agile Testing´ (Crispin**[JL1]**). and ´More Agile Testing´ (Crispin**[JL1]**).

Before implementing Agile Testing and using the BI development and testing process in an agile environment, the following checklist could help with the thought process of implementing:

- Start with the business information needs to provide context for scope,
- Iterations should be time-boxed,

- Stress on data discovery through the requirements and design phase,
- Validate the BI Architecture and get approval on the proof of concept,
- Data validation and verification[9] should be completed for each development iteration,
- Use flow charts or diagrams to explain the BI process along with some documentation,
- Any change that will be deployed to production should be thoroughly tested in regression environment,
- Environment needs for the different DevOps teams/ CI/CD pipelines,
- Lean solutions to add like Monitoring, tests in production, Build-in Quality (especially in a D&A environment where Dataflows can be unpredictable),
- Have a formal change control; this will minimize the risk as all changes have to be approved before it goes into production.

2.4 Risk-based Testing in a BI & DA Environment (LO-2.7, LO-2.8, LO-2.9, LO2.10)

A risk is something that might lead to a future negative consequence. Based on the probability that something occurs and the impact when that events happen. Risk based testing is an approach for testing that is trying to reduce the level of risks on the product by informing stakeholders of the status of these risks. For a description of risk-based testing a reference is made to (ISTQB**[IS1]**).

A tester is able to create a risk analysis using a risk-based approach. During risk analysis, quality characteristics models like ISO/IEC 25010 are being used. Most of these models are focusing on the quality of the product itself, which is fine, but for BI & DA environments additional quality characteristics should be applied. There are specific information and data quality characteristics that are very valuable to be considered. In chapter 5 a more detailed description of these quality characteristics is given.

A tester that is involved in setting up a test strategy and risk-based testing approach should have understanding of these characteristics and specific risks in a BI & DA environment. And how risk-based testing a support BI testing. For instance, localization issues, or format issues (e.g. amounts) between different data sources or countries.

When a tester resides in an agile environment, it would be expected that during the sprint planning meeting the tester thinks about risks and risk-based testing. Therefore, the tester should be capable and have understanding of risk-based testing, and in this case specifically in a BI & DA environment.

2.5 Testing Roles and Skills (LO-2.11, LO-2.12, LO-2.13)

The ISTQB covers the roles of tester and test leader (ISTQB**[IS2 par 5.1.2]**) and describes their tasks in a generic manner. Whereas the organization of testing in a BI & DA environment is quite similar, its landscape is complex and testing this requires a great variety of skills.

[9] Data verification is a way of ensuring the user types in what he or she intends, in other words, to make sure the user does not make a mistake when inputting data. ... Validation is about checking the input data to ensure it conforms with the data requirements of the system to avoid data errors.

Think of skills in the following areas:

- Querying (e.g. SQL, NoSQL)
- Infrastructure (e.g. system engineering)
- Data Modelling
- Data Visualization
- Data Integration and Reporting Tools
- Test Data generation
- Data Quality
- (Inter)nationally acknowledged legal frameworks regarding Privacy, e.g. General Data Protection Regulation (EU) 2016/679 (GDPR).

When addressing roles and responsibilities, using generic terms like “tester” or even “data warehouse tester” is dangerous, as they differ in different organizations and projects. To cope with this, the following roles are distinguished as an example:

1. Reporting Tester
2. ETL Tester
3. Data Migration Tester
4. Data Quality Tester

Depending on the organization and type of project, different knowledge and skills of the tester will be required. The spider graph in Illustration 4 provides a strong *suggestion* on the knowledge levels needed for each role and (project) environment. The descriptions that follow below will highlight some of the characteristics of such testers.

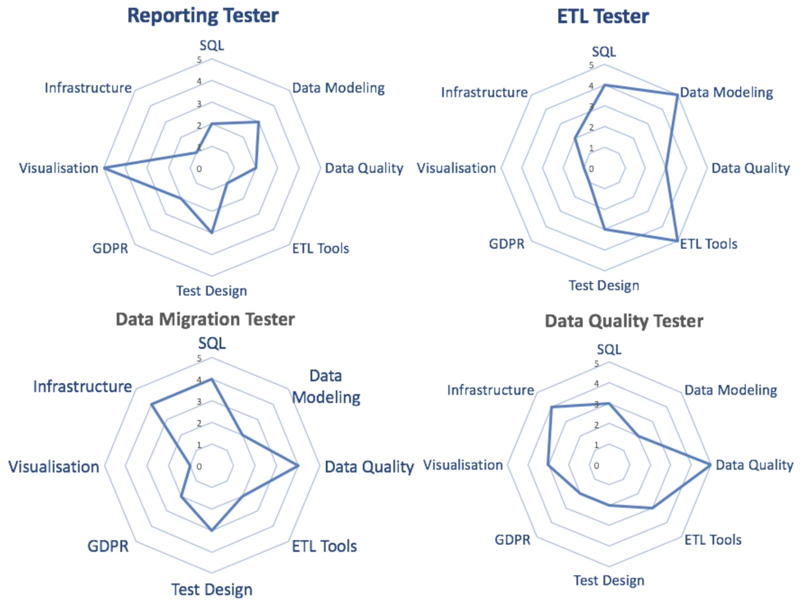


Illustration 4 Testing roles and their specific skills (Source: DaU, 2020)

Reporting Tester

The Reporting Tester (or: “Reports Tester”, “BI Tester”) will focus mainly on the end user product, interacting with the developer of reports and dashboards and with the business. This tester will excel in knowledge of business standards, subject matter expertise and industry knowledge (for instance, Health Care, Finance or Marketing). Most Reports Testers can be found close to the Business.

ETL Tester

In general, ETL Testers will have knowledge of logical database design, ETL-processing tools and their querying skills will enable them to create comparisons between source and target tables, incorporating aggregations and transformations according to (general) design rules and technical designs like Source-to-Target Mappings. Often, ETL Testers will have knowledge of the underlying infrastructure (databases) so they can design test cases accordingly. ETL testers will also focus on robust processes and availability of the system/data. Most ETL Testers can be found close to the Developers.

Data Migration Tester

Like the ETL Tester, Data Migration Testers will need to show that data is moved to a different system in correct and complete fashion. The target may be a different technology and/or from on premise to a cloud- based solution. This requires additional knowledge about storage of data, data types, performance characteristics etc. As the migration is often planned as a one-off activity (one migration after a few trials), robustness is often of lesser interest. Data Migration Testers are likely to be found in Task Forces that perform similar data migrations for different customers, using elaborated re-usable scenarios.

Data Quality Tester

Data Quality may be addressed as such an important aspect that Data Quality Testers are being introduced. DQ Testers will excel in measuring the expected and perceived data quality in the system, at source systems (OLTP) or the Data & Analytics landscape. In order to analyze results, such roles require knowledge of the industry, behavior of the underlying infrastructure and tools to perform the analysis. In the field, Data Stewards may be asked to perform this testing role next to their other activities.

2.6 Set-up a test-strategy and approach in a BI environment (LO-2.14)

Because a BI & DA environment got many aspects in it, like transformation, complex infrastructure, reports and a diversity of database solutions it is important for a test manager (or tester, when there is not a test manager) to think about setting up a test strategy and approach to have a good understanding of the different test aspects of testing in a BI & DA environment. You should setup a test strategy and approach based on the BI & DA environment. This should include an explanation of aspects like functional- and non-functional testing and other BI or DA aspects based on the specific areas in the Illustration 1. In this approach elements of test techniques, quality attributes (non-functionals and data), what need to be tested and to which extend (risk based) testing will be executed.

3 Test Techniques

In this chapter, a summarize of the test techniques is described that can be useful for a tester to apply.

Keywords

Syntactic test, Semantic test, Equivalence Partitioning, Boundary testing, Decision Table Test, Data Combination Test, Coverages

LO-3.1	K3	Explain the Syntactic test technique and be able to apply them
LO-3.2	K3	Explain the Semantic test technique and be able to apply them
LO-3.3	K3	Explain the test technique Equivalence Class Partitioning and be able to apply them
LO-3.4	K3	Explain Boundary Testing and be able to apply it
LO-3.5	K3	Explain the test technique Decision Table test and be able to apply it
LO-3.6	K2	Explain the test technique Data Combination test and be able to apply it
LO-3.7	K2	Explain why different coverages are required within test techniques

There are several books that explain the test techniques in a very detailed way. Especially for experienced testers, this information can be common knowledge. From the perspective of a BI analyst or –consultant this information could be rather new. For this last group, we refer to (Koomen**[TM1]**) and (ISTQB**[IS2]**) to have a broad understanding of how these techniques work and can help you in a BI & DA Environment.

3.1 What are test techniques

Test Techniques are methods to derive test cases from source documentation in a structured way. Test techniques are useful to help the tester to have a good understanding of the source documentation, find defects in the source documentation, and to set-up test cases in a structured and most efficient way. Test techniques and combining techniques help the tester to setup and define test cases so testing can be executed in the most sufficient and efficient way.

3.2 Different techniques (LO-3.1 to LO-3.7)

There are a lot of different techniques that exist in the world. There are several test methodologies like (Koomen**[TM1]**) and (ISTQB**[IS2]**) that described techniques, like:

- Syntactic Testing
- Semantic Testing
- Equivalence Partitioning
- Boundary Value Analysis
- Decision Table Testing
- Elementary Comparison Testing
- Data Combination Test
- Checklists Based Testing
- Pairwise Testing
- Experience Based Techniques, like Error Guessing and Exploratory Testing

- Coverages, like Statement- (SC), Decision- (DC), Condition Decision- (CDC), Multiple Condition Coverage (MCC) [ST1]

For a detailed description of these techniques and the usage of these techniques a reference is made to (Koomen**[TM1]**), (ISTQB**[IS2]**)**,** (Veenendaal**[TT1]**) and (Black**[ST1]**). It doesn't mean that other techniques that are available are not useful. Most important for a tester is that a tester is knowledgeable of all kinds of techniques that are useful in a BI & DA environment and able to practice and apply these techniques. The techniques mentioned above, are techniques useful in all kinds of areas of software development. In chapter 4 test techniques are described that could be useful in specific areas of a BI & DA Environment.

3.3 Mapping techniques on a BI & DA Environment (LO-3.1 to LO-3.7)

Based on illustration Illustration 1 A typical BI environment there are different tests techniques defined on the areas of the model. The table below describes a common use of techniques to those different areas. This doesn't mean, that depending on the situation techniques could be used on all other areas, but this table focuses on the typical techniques for a certain area:

Area	Typical test techniques used to test the area	Typical documentation used as source for testing
Databases (DWH, Staging, Datamarts, data lake)	Syntax, Semantic	functional specifications (of source systems), ERD, users' subject matter expertise (SME)
OLAP	Syntax, Semantic, Decision Tables, Coverages, Pairwise, Checklists, Business Process Test	Business processes, functional and technical specifications
Reports/Dashboards	Syntax, Semantic, Checklists, BVA, EP	Functional- and technical specification document, mockups, company standards
Data Models	Experience Based, Checklist	Logical Data Models (in modelling tools)
ETL	Syntactical, Semantic, Equivalence Partitioning, Boundary Value Analysis, Coverages, Pairwise, Decision Table	Business rules, Source Target Mapping
Internationalization/Localization	Syntax, Semantic, Checklist	Source Target Mapping, Functional Specification Document, Standards (e.g. ISO), SME
Data Quality	Data Profiling, Experience Based	(Production) Data, Data Dictionary, (company and industry) Standards, SME

Table 2 3.3 Mapping techniques on a BI & DA Environment

This table shows the typical techniques used and could be used in different areas of the BI & DA environment. Coverages for example, got quite some different approaches that could result in less or more test cases. Depending on the risk of the subject area or the information that the data is used for, the tester can decide to use are more extensive coverage. This table is therefore helpful in the process of setting up a risk-based test strategy and to define the approach of testing.

4 BI Testing

In this chapter, terminology is explained about testing in a BI & DA environment like reports, data mining, and testing transformations and how to apply test techniques.

Keywords

BI Reports, Queries, Dashboards, Self Service BI, OLAP, Data Modeling, Data Mining, Data environment, ETL, Transformations, Aggregations, Conversions, STM, Business rules, Completeness testing, Minus/Except, Intersect

LO-4.1	K2	Explain the testing of BI Reports and Dashboards, including its place in the data environment
LO-4.2	K1	Recall OLAP, Slicing and Dicing
LO-4.3	K1	Recall the description of Data Modeling and normalization
LO-4.4	K1	Recall the description of Data Mining, including Testers' Tasks
LO-4.5	K1	Recall common linguistic confusions among testers and data scientists
LO-4.6	K2	Recall the aspects of completeness testing
LO-4.7	K2	Recall common ways to compare dataset, by using visual aids, file comparison tools, database comparison tools and SQL statements.
LO-4.8	K3	Explain Comparing Datasets using database SET operators MINUS/EXCEPT and INTERSECT and be able to apply these
LO-4.9	K2	Recall the aspects of transformation testing

In a BI & DA environment, test types **[IS1]** can be executed on several test levels (ISTQB**[IS1]**). Mostly, in a BI & DA environment the difference between the test levels is not so great. Based on the traditional test types like performance etc., there are some specific ones in a BI & DA environment:

- Reports- and Dashboard Testing
- Testing of OLAP cubes
- Testing of Models
- Testing in Data Mining
- Completeness Testing
- Transformation Testing
- E2E Testing

4.1 Reports- and Dashboards Testing (LO-4.1)

In Chapter 1 reports and dashboards are introduced as a way to visualize data, to provide insights to our business colleagues. When discussing testing of reports, think about queries (ad-hoc), reports, dashboards, self-service BI, OLAP (see paragraph 4.2) and data mining (see paragraph 4.4).

In general, testing reports and dashboards leans heavily on the use of checklists **[ST2].** These checklists will focus, amongst others, on the following aspects:

- Access
  - Who is authorized to see the report and who is not? Is any Role Based Access applied?
  - Can certain fields be viewed by some users only, for instance, based on his/her role? Should these values be (partially) scrambled or hidden?
  - Can the report be saved and shared with other users? With authorized users only or with anyone?
  - Is the report accessible (and still readable) on mobile devices?
  - Can the report be printed and is its lay-out as expected?
- Field mappings
  - Are all fields mapped to the right fields in the source (database)?
  - Are the naming conventions as expected?
  - Are the transformations and aggregations as expected?
  - Are the values logically sorted and/or according to the design?
  - How should records with NULL values be treated?
- Visuals
  - Is the lay-out (incl. font, color, etc.) as expected, when looking at company standards?
  - Are the widths of columns sufficient to store the widest names and numbers in the source – and what if the numbers are totaled?
  - Are graphs implemented as expected, including the right colors, captions and legends?
  - Can visuals be understood by the color blind?

Many of the aspects mentioned above can be considered as syntactic- and semantic testing. Checklists are easily adapted to the actual situation at the client's site and in general they can be easily taught to new colleagues.

4.1.1 Data for testing Reports

Most reports will eventually get used by business users, so for acceptance testing it is important to provide test data that can be recognized by users and therefore is similar to production data. When the report is being developed on an existing Data Mart and personal data is involved, it may be the best to have (a sample of) the Data Mart pseudonymized (see chapter 6.4 regarding privacy regulations).

If a Data Mart does not yet exist, it is recommended to have the report built upon a *virtual* Data Mart, created from (database views on) data in the source systems and/or the data warehouse. Due to this work around, performance may be poor in this phase, but it will allow for a great deal of the necessary functional testing. Performance testing will be performed in a later stage, on the real Data Mart. The development process of the virtual Data Mart itself will probably lead to finding design flaws – which can then be tackled early.

At the stage of systems testing, when testing technical aspects like mappings and formatting, it may be best to define a small synthetic test set.

4.1.2 Query Skills for Testing Reports and Dashboards

Testing reports and dashboards requires skills regarding the presentation tool itself, and skills for querying the data in the underlying systems (e.g. the Data Mart). In general, as Data Marts are designed for "easy" querying, this leads to the need for basic query skills like SQL. The tester must be able to compare the presented values to the database entries. Usually, no complex JOIN operations need to be performed in this part of the DA environment.

4.1.3 Testing Ad Hoc Queries

Sometimes the business will require an *immediate* answer based on company data that was not (yet) disclosed in an orderly fashion (e.g. through reports or dashboards). This is when Ad Hoc Queries may be used. Ad Hoc Queries are often (SQL) database queries written by developers, creating a one-off product that provides the necessary data to the end user.

As the sources systems may not be designed for reporting purposes, the result will rely heavily on the knowledge of the underlying data and the way source systems and business processes work. Also, as the query will often not be required to be future proof, the coding will likely to be more complex (and harder to read). Extensive (SQL) coding skills from the developer and the tester(s) are needed.

Ad hoc queries can be tested on at least two levels. The technical level: is the query selecting the right data (using the correct joins, tables etc.). The functional level: is the query giving the right data that the user asks for (is it the right information delivered). During design, development and testing, frequent contact between the subject matter expert, developer and tester is heavily recommended.

4.2 Testing of OLAP Cubes (LO-4.2)

4.2.1 Drill Down, Drill Through, Slicing and Dicing

Common functionality that is offered by OLAP Software, is to drill down, drill through and to slice and dice the data. By comparing results of testing queries (by ways of SQL, MDX) with the presentation in the OLAP Cube, testers can validate that correct values are presented. Important validations are those of the aggregations: will regions, product groups, timelines be shown in a correct manner? What is the lowest grain of the values and should this be shown? What is the result of renaming country names, what will happen when very large and very small sets exist? Are dates and timelines (SCD) and codes/decodes implemented in a correct manner? What are default values for NULL?

There is more information about this subject, for example Kimball group provides a series of flaws that emerge in data modeling, amongst others: grain, codes, hierarchy handling, date dimension issues, surrogate key issues, SCD strategies **[KIM1].** SQL skills needed to test OLAP Cubes will include basic aggregation skills like SUM and GROUP BY functions.

4.2.2 Self Service BI

Tools that are used for OLAP and Reporting are grouped among the "Self Service BI (SSBI)" tools, tools that provide querying features for (business) end users. So, testing methods for "off the shelf software" are required as well. Labels, meanings and references in data sets for "SSBI" must be as transparent as possible for end users' purposes: where developers and testers tend to read data definitions, many end users will not invest time in learning the exact details and differences between the presented data objects.

4.2.3 Performance Testing of OLAP Cubes

Copying data from relational databases to in-memory OLAP cubes, or loading the data, will take up time. Probably the time needed will increase varying on the moment of the day (are business users and other processes using underlying systems?) and the amount of data available (data sets will grow over time?). Therefore, performance tests may be introduced to validate that the cubes can be loaded within the time window that is made available. Questions will arise on who can load cubes (end users?) and when can cubes be loaded (batch window, ad hoc). Also, requesting data from the cube will take up time. Performance of the response times resembles testing of (generic) front-end tools. For this purpose, regular test execution tools for front ends can be applied. Tools like Tableau even have built in performance measuring.

It is a good practice to have dba-consultants early involved who look after efficiency of the queries and database-schemes. Then lot of performance issues can be prevented.

4.3 Testing the Data Model (LO-4.3)

4.3.1 Testing the Data Model

Static testing of the data model may consist of reading design decisions and reviewing logical data models. It is important to recognize "implied" design rules, when modeling rules are applicable like dimensional modeling[10] and/or data vault modeling[11].

When Logical Data Models (LDM) are implemented on the actual infrastructure, designers and developers may choose to apply Physical Data Models that are different from the LDM. Effects of changes should be considered when writing test plans. Especially adding Indexes[12] and Partitions[13] may influence performance of reading and writing the data.

Testers and designers may create test scenarios focused on the growth of the data sets and its performance in the near future (see chapter 4.3.2).

- When starting with an empty data warehouse, are empty tables allowed?
- Are scripts needed to create a valid starting point for loading the initial load and the following increments?
- How much time does the initial load take?
- Will the system provide information (warnings, error logs) when unexpected loading patterns and occur, for instance a "wrong loading order" of data sets?
- Who has access to this audit trail or logs?

[10] Dimensional Modeling is a data structure technique optimized for data storage in a Data warehouse

[11] Data vault modeling is a database modeling method that is designed to provide long-term historical storage of data coming in from multiple operational systems

[12] An index is a method to track the performance of some group of assets in a standardized way

[13] A partition is a section of a storage device, such as a hard disk drive or solid-state drive

- Does it hold any personal identifiable data (see chapter 6.4)?

4.3.2 Testing Implementation of History (Data Lineage)

Data lineage states where data is coming from, where it is going, and what transformations are applied to as it flows through multiple processes. It helps understand the data life cycle and time aspects. It is one of the most critical pieces of information from a metadata management point of view (Allen**[MA1]).**

In both static and dynamic testing, testers should stress testing time aspects:

- When loading data into the data model, how will the order of loading data sets influence the results?
- Will starting and ending timelines be correct? Will no data "disappear" or stored twice?
- Validation of SCD Types (using the Source to Target mapping): which tables have what SCD type?
- Can some tables be changed periodically, for example, only once a month or year?
- When loading multiple sets at the same time, do timing issues occur?
- How to test Timing triggers in different tables, or different levels or different related databases in near-real-time? E.g. When the system starts loading for 8 hours at 12 PM, but the DataMart needs to be loaded at 7am, because users need their reports at 8am. What happens?
- Are all start dates and end dates in every relevant table affected?

4.4 Testing in Data Mining (LO-4.4, LO-4.5)

As stated in Chapter 1, Data Mining is leaning heavily on automated discovery of patterns in data and will require in-depth knowledge of statistics and Data Science.

The data science process partially resembles software development, at those points data and analytics testers can successfully contribute to the development of Data Mining deliverable. This can for example be done by reviewing designs, challenging assumptions, testing the underlying infrastructure.

4.4.1 Beware of sector specific terminology

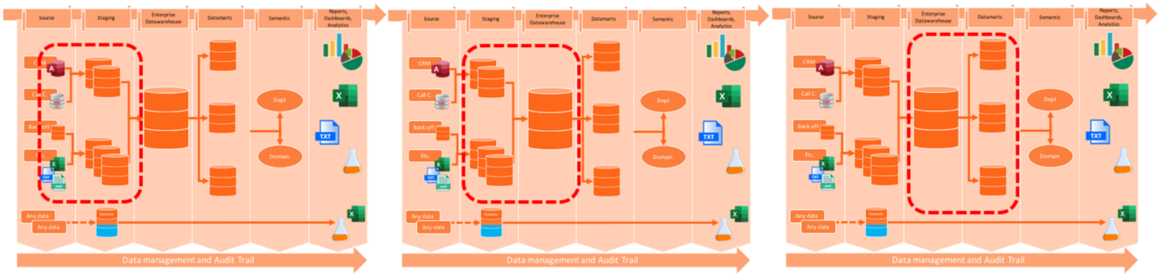
Due to different naming conventions in the world of data science and software development, linguistic confusion among testers, developers and data scientists may arise. In Data Science terminology, "performance of an algorithm" refers to the quality of its predictions, not to its speed. Also, a "test set" has a different meaning to a tester than to a data scientist. The latter will speak of training sets, validation sets and a test set or holdout data set, in the process of creating and testing the (machine learning) algorithms.

For a tester, regression will mean that errors are being introduced when new versions of the software are deployed. For a data scientist, regression is a statistical term, used to describe the strength of relationship between two or more variables.

4.5 Completeness Testing (LO-4.6)

Completeness testing is required to ensure that all relevant records are transferred from the source to the target and that the contents (format, precision) of each record (row) and value (column) is still correct.

Other ways to verify completeness may include checking the audit trail or executing other reconciliation procedures. This is relevant to both the ETL-process and the Reporting activities.



One of the methods for completeness testing is to count records, for instance, by the SQL SELECT COUNT statement. In most cases, this is not sufficient: counting will only prove that the number of records is as expected, not that the values in the cells (records) are correct – or data inadvertently shifted over records.

4.5.1 Common ways to compare data sets (LO-4.7)

For testing the ETL process, data sets need to be compared frequently. Depending on the size (volume) and type of data, different approaches and tools may be applied. In general, testing needs to validate that all relevant rows and columns are being copied and no data is either lost or wrongly duplicated.

4.5.1.1 *File Comparison Tools*

File comparison tools will support testers when loading two files and reporting in which rows or columns differences occur. For example, by visually showing this (on-screen) to the tester or by generating a log with differences. Use of file comparison tools may become difficult in a Big Data environment, due to for example long loading times, insufficient memory space or lengthy (visual) analysis by human testers. For smaller data sets, like reference tables and sets up to a few hundred rows, the use of file comparison tools is recommended as the tools are easily available.

4.5.1.2 Data Quality Tools featuring Comparison Functionalities

Data quality tools (see chapter 5) often provide features that enable file/table comparisons. Of course, this is not the only function of tools, they will also inform their users on many aspects of the data quality like its frequencies and "odd" situations in the data sets.

4.5.1.3 Database oriented comparison

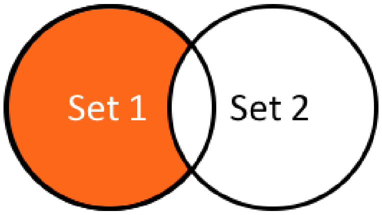
Data sets that are stored in (relational) databases can be compared using SQL statements, in order to verify that all rows were copied, and all content is in the expected place. The most basic SQL statements, unfortunately, often require lengthy and complex JOIN constructs. The time and the needed skills may not be sufficiently available. Some database systems will offer specific SET operators that make comparison easier.

4.5.2 Compare datasets with database SET operators (LO-4.8)

Some database environments support the MINUS (or EXCEPT) and INTERSECT query. Those queries (officially called SET operators) can facilitate a much easier comparison of source vs target tables. The process is to execute a source MINUS target query and a target MINUS source query. Check if there are for duplicates – and missing rows.

The MINUS operation combines result of two SELECT statements and returns distinct rows which belong to first set of results only. A MINUS or EXCEPT query looks as follows:

```
SELECT column1 [, column2 ]
FROM table1
[WHERE condition]
EXCEPT
SELECT column1 [, column2 ]
FROM table2
[WHERE condition]
```

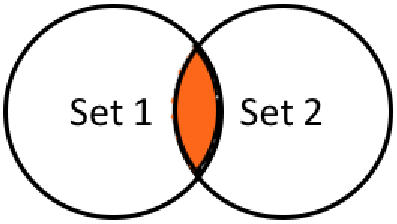


The SQL INTERSECT clause/operator is used to combine two SELECT statements but returns rows only from the first SELECT statement that are identical to a row in the second SELECT statement. This means INTERSECT returns only common rows returned by the two SELECT statements. The basic syntax of INTERSECT is as follows.

```
SELECT column1 [, column2 ]
FROM table1
[WHERE condition]

INTERSECT

SELECT column1 [, column2 ]
FROM table2
[WHERE condition]
```



MINUS queries must conform to the following rules: First, the number of columns in the select lists of both queries must be the same. Second, the order of the columns and their types must be comparable. With very large (broad) tables it can be efficient to compare the most relevant columns only, for example, by creating VIEWs on the tables first and then perform the MINUS queries on these VIEWs.

Note that in the standard MINUS query implementation, MINUS queries need to be executed twice (Source-to-Target and Target-to-Source). This doubles execution time and resource utilization. MINUS queries are processed on either the source or the target database, which can draw significantly on database resources (CPU, memory, and hard drive read/write). In most cases, as the environments have to be set up to provide for these kinds of queries, it should not be a problem. But to assure, always contact your Database Administrators in order to avoid performance issues.

4.6 Transformation Testing (LO-4.9)

In Chapter 1 the Source-to-Target Mapping document was discussed in which you will find a description of the required transformations. For transformation testing it is important to understand the different forms of transformations that can be described in the STM.

4.6.1 Examples of Transformations

An example of a simple transformation in the ETL processes is the concatenation of values. Columns in the data mapping document are sometimes combined in a single target column. Here, testers will need to validate at least that this simple concatenation will function properly, even when NULL's/empty values exist in the source table and upper- and lower-case characters are used.

Aggregations can be filed under Transformations as well. Common aggregations are:

- periodic values, for instance Hour > Day > Week > Month > Quarter > Year.
- geographical values, for instance City > Province > Country > World region.

Other forms of transformation, are decoding conversions for integration of data or easier readability for end users:

- M/F and 0,1,2 ➔Male, Female, Unknown
- EUR, €, Euro ➔ EUR
- 1.000.000,= / 1,000,000.-➔ 1000000,00
- 01-10-2012, 1-OCT-2012 ➔ 20121001000000.

A tester is required to have in-depth knowledge about the datatypes available in the database system and the data integration tools used. Look for tricky transformations and point out risky differences in character code settings. When designing test cases, data profiling can be performed in order to discover which actual values in the source tables exist, that need to be transformed in the design and the implementation of the ETL.

4.6.2 Applicable Test Techniques

Dependent of the transformation different techniques could be used. When data is transferred for column A in source to column B in target without any transformation the semantic testing will be a helpful technique. When there is a business rule with a transformation (for example source value "SALES" to target value "S") then several techniques can be used, for instance Equivalence Partitioning, Boundary Value Analysis, Data Combination or Decision Table Test are all good techniques that can be applied. Which one you choose, will depend on the level of risk identified during the risk analyses. You can expect that a decision table can result in more test cases than an equivalence partitioning. Also, the coverage techniques from (Koomen**[TM1]**) can be applied here.

4.7 E2E testing

E2E testing is an abbreviation of end to end. Basically, it is meant a test that starts at the complete beginning of a process, the trigger or cause until the processes get to the finish state. The same applies for a BI & DA environment. The process starts at extracting the data from the source, then data flows via the staging, data warehouse, data marts to the end result, a database, reports or dashboard. E2E testing in a BI & DA environment can therefore be compared to a complete integration test from start (extracting the data) up and until the end results.

During E2E testing in a BI & DA environment that is therefore also the definition that is being used. Testing of the integrated process from extracting the data from the source until it receives their end state. Typical aspects that are being tested in this process, is infrastructure, interfaces, and consistency, traceability and completeness of the data flows.

Complexity within this process is the timelines, to get insight in the different data flows, but also the timings of for instance the different sources and especially when data from different sources must be in sync with each other.

5 Data Quality

In this chapter, the terminology concerning Data Quality is defined and explained and how to measure the level of Data Quality.

Keywords

Quality, ISO/IEC 25010, ISO/IEC 25012, DAMA, data profiling, SQL, outlier detection, datatypes, big data

LO-5.1	K1	Understand that there are different definitions of Quality and perspectives
LO-5.2	K2	Recall the main quality characteristics according to ISO/IEC 25010 and explain the shortcomings when it comes to Data Quality
LO-5.3	K2	Recall the definition of Data Quality according to DAMA and apply each of data quality characteristics to a given data set
LO-5.4	K1	Recall the definition and reasons of Data Profiling
LO-5.5	K1	Recall the most common (single column) data profiling tasks
LO-5.6	K1	Identify basic SQL statements for Data Profiling
LO-5.7	K3	Apply data profiling tasks to an example data set incl. using the correct terminology
LO-5.8	K1	Recall benefits of using Data Profiling Software
LO-5.9	K1	Explain the impact of Big Data on the Data Profiling process

5.1 Quality (LO-5.1)

There are many definitions of quality, the following bullet-list represents four of them. All to a certain extend explain quality in their context. It is important to understand the different viewpoints of the different definitions, as this is also the authors' view on quality in the context of data.

- William Deming, an American engineer, statistician, professor, author, lecturer, and management consultant has defined quality as follows: *"Good quality means a predictable degree of uniformity and dependability with a quality standard suited to the customer."*
- The American Society for Quality Control Standards Committee defined quality as: *"The totality of features and characteristics of a product or service that bear on its ability to satisfy stated or implied needs"*
- The European International Standards Organization defined quality in the ISO/IEC 9000 as: *"The degree to which a set of inherent characteristics, of a product or service, fulfill requirements".*
- Joseph Juran build theory's and methods of Total-Quality Management and defined quality as: *"Fitness for use. Those product features which meet the needs of customers and thereby provide product satisfaction. Freedom from deficiencies".*

5.2 Quality Characteristics (ISO/IEC 25010) (LO-5.2)

The international standard ISO/IEC 25010 provides us with a model focused on the quality of computer systems and software. The standard describes product quality using the following eight main characteristics:

1. Functional Suitability
2. Performance Efficiency

3. Compatibility
4. Usability
5. Reliability
6. Security
7. Maintainability
8. Portability

Data and Analytics systems can be regarded as a form of computer systems and software too, so the characteristics will help us to analyze the (desired) quality level of the system. Depending on the focus, for instance extracting data from sources vs reporting, other quality attributes (see chapter 5.3) can be more useful, see chapter 2.4 on Risk Based Testing too.

However, establishing the level of Data Quality may prove difficult using the ISO/IEC 25010 model. Data Quality may be part of "Functional Suitability", by linking to its sub characteristics: Functional Completeness, - Correctness and – Appropriateness. However, it does not (yet) provide us with sufficient means to measure data quality hands on, which forces Data Analytics professionals to explore other options for measuring data quality.

5.3 Data Quality Characteristics (LO-5.3)

When discussing data quality dimensions (or: characteristics, attributes) even data quality professionals see themselves confronted with a variety of definitions of data quality dimensions. To communicate about data quality in a clear and concise way, professionals were forced to develop their own set of data quality characteristics in the past.

Lately, standards are being developed like the IPS Model (Bouman**[EB1]**), the ISO/IEC 251012:2008 (ISO**[ISO]**) and the DQ Characteristics as presented by the DAMA Working Group on Data Quality (DAMA **[DAM]**). We will discuss the latter two in short, encouraging you to read more into it.

5.3.1 Data Quality according to ISO/IEC 25012:2008

The quality models of ISO as presented in the ISO/IEC 2501x series are extended with a specific model for Data. The data quality model defined in ISO/IEC 25012 outlines fifteen DQ characteristics and regards these as "inherent" and "system dependent" data quality:

	Data Quality	
Characteristics	**Inherent**	**System Dependent**
Accuracy	X	
Completeness	X	
Consistency	X	
Currentness	X	
Accessibilty	X	X
Compliance	X	X
Confidentiality	X	X
Efficiency	X	X
Precision	X	X
Traceability	X	X
Understandability	X	X
Availability		X
Portability		X
Recoverability		X

Table 3 ISO/IEC 25012

5.3.2 Data Quality according to DAMA

DAMA International is a not-for-profit, vendor-independent, global association of technical and business professionals dedicated to advancing the concepts and practices of information and data management. A working group of DAMA created a concise and useful set of data quality characteristics that can be well used in the BI & DA environment to establish the actual data quality.

Data Quality (DQ) as stated in the DAMA International, Data Management Book of Knowledge "Refers to both the characteristics associated with … and to the processes used to measure or improve the quality of data." Data is considered high quality to the degree it is fit for the purposes data consumers want to apply it. […]. The working group provides a set of six dimensions and provides measures for each of the dimensions (Dama**[DAM]**).

Testers will need to recognize the different dimensions and apply the measures:

1. Completeness (is all data actually stored in the system)
2. Uniqueness (data is stored only once; no duplicates exist)
3. Timeliness (difference in time between the situation in the real-world vs the recorded data)
4. Validity (conformity to syntax rules)
5. Accuracy (correct and precise description, in context of real-world vs the records)
6. Consistency (no differences when data is stored in multiple places)

Testers in any data driven domain will need to know and understand the characteristics and know how to apply these in order to define, calculate and interpret data quality characteristics. (Dama**[DAM]**) describes these elements in more detail, including the calculations used and how to interpret the results of it.

5.3.3 Testing of DQ Characteristics

In order to provide a clear report on the data quality (and its influence on the system, End-to-end) it is important to select the most relevant characteristics and to define a level of required quality. This is typically done in the product requirement analysis (PRA) or shortly after the first analysis of source systems data.

Setting up measures to test the quality and executing these tests, repetitively, will help to inform both business and development teams about potential problems that may be difficult to find afterwards, downstream, in the environment. Typically, after this phase important choices will be made regarding the robustness of the system, to what extent will the code need to be able to process unexpected data and when it is confronted with erroneous data, will it log such situations and continue processing or abort processing altogether? Depending on the choices made, this may lead to *adding* test cases with Synthetic Test Data testing the robustness with "illegal" situations (as these faulty situations may not yet be available in the production systems).

5.4 Data Profiling (LO-5.4)

Data Profiling is the process of examining the data available in an existing data source […] and collecting statistics and information about that data (Johnson**[TJ1]**).

Data profiling will provide important insights to designers, developers and testers in the Data Analytics domain. Checking whether values really "never occur", what are the actual domains in which values fall, what seem to be the character settings in source systems, etc. These will lead to better specifications and/or more and improved test cases.

Testers will apply data profiling in order to:

- "Shift Left"
- Facilitate better Modeling
- Facilitate better Data Integration
- Improve and extend Test Cases
- Improve Coverage Levels of Tests
- Enable testing of Error Handling and Audit Trails

Other phases will benefit from data profiling like:

- Efficient Database Design
- Data Clean(s)ing process
- Reverse engineering

5.4.1 Common Data Profiling Tasks (LO-5.5)

Below a summary of common Data Profiling tasks is shown:

Category	Task	Description
Cardinality	Num-rows	Number of rows
	Lengths	Measurements of value length (min, max, median, average)
	Nulls	Percentage of NULL values
	Distinct	Number of distinct values (also called: Cardinality)
	Uniqueness	Distinct Values/Number of Rows
Distribution	Histogram	Frequency histograms
	Constancy	Most frequent value / number of rows
	n-tiles	Grouping numeric values in equal groups, e.g. Quartiles
	1st Digit	Check for Benford's Law
Patterns, Data Types and Domains	Basic Type	Generic data type, e.g. numeric, alphabetic, alphanumeric, date/time
	Data Type	Database specific datatypes, e.g., VARCHAR, DATE, CHAR(10)
	Size	Maximum of digits in a numeric value (precision)
	Decimals	Maximum of decimals in decimal types (precision)
	Patterns	Value Patterns, e.g. "Aa9…"
	Class	Semantic, generic data type, e.g. Code, Indicator, Text, Date/Time
	Domain	Classification of a semantic domain, e.g. credit card, first name, city

Table 4 common Data Profiling tasks

Data Profiling will look into the way that data is distributed in the system. Data profiling in order to discover errors and questionable situations can be performed either by a designer of by a tester. The effect of a data profiling assessment is that design decisions can be adjusted, additional test cases can be created and small but effective test sets can be derived from the complete (production) data set.

5.4.2 Basic SQL statements for Data Profiling (LO-5.6/5.7)

When performing data profiling on a relational database, basic and manual SQL statements can be used.

Two examples of common SQL queries[14]:

```
*/ INDIVIDUAL UNIQUE VALUES /*
SELECT COUNT (X) FROM TABLE
GROUP BY X;
```

```
*/ PROPORTION OF UNIQUE VARIABLES /*
SELECT CAST(COUNT(DISTINCT X) AS DECIMAL)/COUNT(X)
FROM TABLE
```

In these examples, only one variable/columns at a time is being profiled – and only for its frequency. In the BI & DA environment many data stores exist, each with dozens to hundreds of tables and each table can contain many columns. Data profiling may become a very time- consuming task. BI-Tools may be used (as they often are available in the BI & DA environment) to help out, or dedicated data profiling tools may be introduced. See par 5.4.4.

As a tester, practical experience with some data profiling tasks or tools is useful to have a good understanding what a tester can benefit of data profiling. Off course, manual querying on a database can help, but tools exist that can speed up this process, and give quick insights and information about databases. The information gathered with the tool can help the tester with understanding the data in the database, and support with developing test cases. Even with just some data profiling knowledge, it is possible to identify possible problem areas with, for example, transformation rules.

5.4.3 Benefits of using Data Profiling Software (LO-5.8)

Using specific tools to perform data profiling activities has some advantages:

Less manual work required
Data Profiling software can help making the data profiling process more efficient. Data Profiling software can be used to automate certain profiling tasks, enabling the team to run data profiling processes whenever they are needed without becoming a burden to the team.

In-depth profiling
Whereas humans might be inclined to save time by skipping (time consuming) profiling actions, tools will run their program and provide all required data profiling information.

Repository of re-usable checks
Using Data Profiling suites will likely lead to a central repository of checks. These can be re-used in tests and even in code, not only in profiling activities. Having all the checks in one place is also beneficial for the quality of the checks themselves as these can be centrally reviewed whenever business rules are added or changed.

[14] In these examples we use Microsoft SQL (T(ransact)SQL) and not Oracle SQL.

5.4.4 Impact of Big Data on Data Profiling (LO-5.9)

The rise of Big Data influences the process of Data Profiling. Looking at the three V's of Big Data (see: *Big Data – LO1.9*) the following impact is seen:

<u>Volume</u>
The sheer volume of data can become a challenge as systems may not be able to profile data that is arriving in real time and in large volumes. Profiling data that is stored in extremely large data stores may become ineffective and inefficient (the cost is too high, the results come in too late).

<u>Velocity</u>
If data is entering the system at high velocity, the overall profile of data may change faster than can be tracked by standard data profiling tools and principles.

<u>Variety</u>
Being used to profiling texts stored in columns and rows (tables) in relational databases, new ways of storing data leave us with challenges. The data itself is different (e.g. videos, x-rays) or formatted in a different way (e.g. XML, JSON). This force us to be creative and find new ways to profile this data.

6 Environmental Needs

In this chapter, the challenges and complexity of using test data and multiple test environments for data & analytics testing is discussed, including privacy and standard compliance rules in modern world.

Keywords

DTAP, test environment, test tools, compliance, data privacy, GDPR, ISO/IEC 27001, anonymization, pseudonymization, de-personification

LO-6.1	K1	Recall the function of a DTAP structure (in a SDLC)
LO-6.2	K2	Explain the necessity to align different DTAP-environments (e.g. for visualization (BI), model development (analytics), data processing (data warehouse), infrastructure and source systems)
LO-6.3	K1	Recall the compliancy standard ISO/IEC 27001
LO-6.4	K2	Recall and explain the necessity for separation of operational and development- and test systems according to information security standards (ISO/IEC 27001)
LO-6.5	K2	Recall and explain the necessity for separation of operational and development- and test systems according to data privacy law and regulations (GDPR)
LO-6.6	K1	Recall the different encryption methodologies for anonymization, de personification and pseudonymization
LO-6.7	K2	Explain the benefits and pitfalls of using production data vs test data

6.1 Test environments according to DTAP (LO-6.1)

DTAP is an approach used in software testing or development to define the different test environments. Each letter of DTAP stands for a specific environment: Development – Test – Acceptance – Production.

The Development environment is mainly used by developers to create and (unit) test their own program or component. Often the developer uses his own pc to test his own work.

When the developer reached the exit criteria or the agreed minimum quality level, the software is transferred to the T, Test environment, where often independent testers are running their tests. Mostly this is a standardized environment that resembles the actual production environment.

When testers reached the exit criteria or the agreed minimum quality level, the software will be transferred to the A, Acceptance Environment where the business user (and sometimes the system manager/administrator) will test the product to validate that it is fit for purpose and get confidence that it meets their expectations.

After successful acceptance testing the software will be transferred to the Production Environment and is mainly used by the business and functional maintenance, and is hardly ever used for testing activities.

The set of environments can be smaller or larger, for instance having multiple development environments in parallel, depending on the demands of the project.

6.2 Multidimensional DTAP in a BI environment (LO-6.2)

A data analytics DTAP will often be more complex (than OLTP DTAP's) because there are several dimensions of test environments that need to be taken into consideration. Here reasons are described for having a multi-dimensional DTAP.

Think of an environment with multiple sources: you can imagine that each source environment will have his own DTAP. Test environments in a data environment are therefore multidimensional and to structure and manage this properly is a very important and not underestimated task.

A typical BI test environment could have several dimensions:

- Source system development (one or more)
- Data warehouse development
- BI (User Interface) development (one or more)
- Eco system (or infrastructure) development

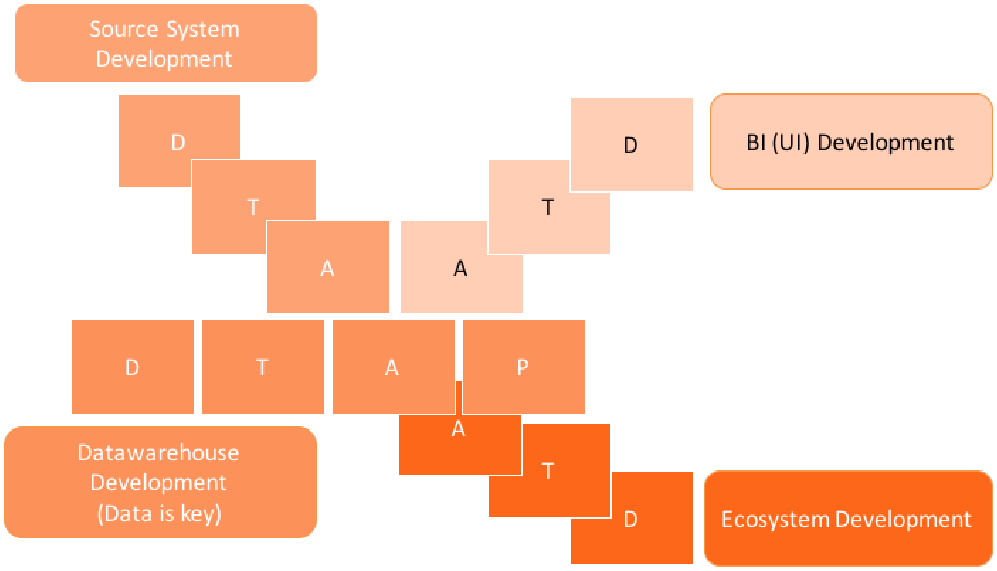

Source System Development
D
T
A
D
BI (UI) Development
T
A
D
T
A
P
A
T
D
Datawarehouse Development (Data is key)
Ecosystem Development

Illustration 5 A typical BI test environment

The complexity depends on the scope, organization, project etc. There are a lot of factors that need to be considered and when the complexity is as complex that all test environments need to be integrated to each other a central coordination activity would be a good suggestion.

Some aspects that are very important to consider and are arguments to have a structured approach and central coordination:

- Same date/time on all environments,
- Data synchronization,
- Time travelling,
- Availability,
- Privacy,
- Identity & Access,

- Synchronization or anonymization.

Also, depending on the needs for tests, different requirements for the test environment are applicable. Some examples of the specific important areas with can result in specific requirements are added:

- Data Processing Test
  - Need for Source Data (profiling),
  - Need for synthetic data,
  - Possibly need for copies of source systems/target systems,
  - Need for the right identity & access.

- Model Development and testing
  - Statistics and Predictive Modelling: need for (full) production data sets.

- Reports testing
  - Need for (near) "real life" data sets,
  - Development of reports separate from Data Mart development,
  - Need for Data Mart-like datasets,
  - Need for the right identity & access.

- Infrastructure testing
  - Hardware vs Software,
  - Upgrading Operating Systems and Analytical tools,
  - Storage, Processing, networking bandwidth,
  - Need for bare metal and/or virtual systems for multiple teams,
  - Local (on premises) vs Cloud based infrastructure.

Data and Analytics Infrastructure often require substantial investment and for instance, Data warehouse Appliances are not bought in numbers like generic hardware (e.g. blades), virtual systems or cost effective and scalable cloud-based solutions. Therefore, the DTAP strategy needs to be addressed as soon as first investment plans are made for the BI & DA environment and test (infrastructure) management needs to be consulted.

6.3 Impact of Security Standards like ISO/IEC 27001 on Analytics Testing (LO-6.3/6.4)

Testing (as well as Development and Change management) require clear written information security policies. ISO/IEC 27001 emphasizes clear rules and policies for the handling of information assets and the engineering process, without stating how this is to be achieved by the organization. ISO/IEC 27002 Provides ways to comply to these requirements.

First, organizations must clarify which data is sensitive and how to handle it. Second, they must explain how the organization buys or creates *secure* software in a *secure* development area. Third, they must state how deploy software into production without any risk for production stability.

Under ISO/IEC 27001 the organization needs to enforce the policies and create evidence for auditing purposes. In other words, a supporting organization needs to be established and employees need to be continuously educated and motivated to act accordingly (Haller**[KH1]).**

6.4 Impact of Privacy Regulations on Testing in the Analytics Environment (LO-6.5)

Privacy is stated as one of the fundamental rights in the Universal Declaration of Human Rights (United Nations). Privacy regulations, like the General Data Protection Regulation (EU) 2016/679 (GDPR) in Europe or the US Privacy Act in America, lead to new requirements to the (use of) the (test) environment. It is very likely that your analytics environment will store personal identifiable information (PII) which is subject to privacy laws and regulations.

These regulations also lead to additional *tests* that need to be performed:

- "the right to be forgotten" gives individuals the right to ask organizations to delete their personal data throughout the whole the BI & DA environment, including data warehouses,
- user logging and monitoring: who is *using* the (personal identifiable) data?,
- encryption techniques.

6.4.1 The Right to be Forgotten – Deleting Data

According to European privacy regulations, data processing parties need to purge (delete) any data that is no longer needed – or when a customer puts in an explicit request to purge his/her data: "the right to be forgotten". This applies to both the development process and the productive stadium.

Make sure that all (personal identifiable) information that was used during development of analytical products, is successfully deleted. If development is outsourced, it is recommended to incorporate this obligation into contracts between parties.

In the production situation, data needs to be deleted when it is no longer needed for the business or when a customer requires deletion. Correct and *complete* deletion of data from production systems, including test systems and -records, backups and data warehouse tables, needs to be tested.

6.4.2 User Logging and Monitoring

In OLTP systems it is common (and often required) to have systems in place that record and monitor activities of its users, in order to have a proper audit trail. Because of the nature of the data in the BI & DA environment (often corporate wide and sensitive data) this requirement is applicable here, too. Activities of (end) users of the analytics products need to be recorded (logged) in order to enable monitoring and analysis by Privacy and/or Data Protection officers. This will require additional tests to be designed and executed, focused on logging and monitoring.

6.5 Encryption methodologies for Anonymization and Pseudonymization (LO-6.6)

For testers as well as for most data scientists, often there is no need to identify individual subjects in the data by using personal identifiable information (like email addresses, social security numbers, credit card details and full-face photographs). According to privacy regulations, this data should not be made available and the data should be de-identified. By encrypting or removing personally identifiable information from data sets, the people whom the data describe remain anonymous.

6.5.1 Anonymization

Data anonymization has been defined as a “process by which personal data is irreversibly altered in such a way that a data subject can no longer be identified directly or indirectly (for instance by using additional data sources)”. In environments like data warehouses, where a lot of data is available to our disposal, there is a risk that subjects can be identified indirectly – so extra caution is required.

6.5.2 Pseudonymization

The process of obscuring data with the ability to re-identify it later is also called pseudonymization. By contrast to anonymization, pseudonymization is defined as “the processing of personal data in such a way that the data can no longer be attributed to a specific data subject without the use of additional information.” *(Article 4(5) of the GDPR).*

Pseudonymization is a data management and de-identification procedure by which personally identifiable information fields within a data record are replaced by one or more artificial identifiers, or pseudonyms. A single pseudonym for each replaced field or collection of replaced fields makes the data record less identifiable while remaining suitable for data analysis and data processing.

Testers will need to inquire if measures need to be taken for anonymization/pseudonymization in the BI & DA environment (for themselves and/or for their colleague developers and analysts) and will need to establish the correct operation of the encryption techniques. In case a tester has doubts or is not sure about privacy regulations or anonymization and pseudonymization regarding testing, they should contact the respective data protection officer (DPO) as they will provide the necessary information.

6.5.3 Complexity anonymization and pseudonymization in a BI & DA environment

Be aware that in general that anonymization and pseudonymization already can be very hard to achieve in the right way in a traditional development project. But in a BI & DA environment if there are for instance ten different sources, both internal and external, where consistency between the sources is required, that synchronically anonymization and pseudonymization can be very hard. This is because data sets are related to one another and keys need to match. Eventually to be successful in a BI & A environment a tester wants, depending on objectives, a consistent data set with data that is related to one another and with keys that match. This is a highly knowledgeable and time-consuming task.

6.5.4 Privacy-preserving Data Mining

In generic data-oriented environments, the rules and regulations stress that the data holding PII should be processed in such a way that it can’t be attributed to a specific data subject without the use of additional information. In Data Mining, the technology is actively linking data (from several sources) and discovering links between them, creating the risk of re-identifying subjects. In order to counter this, algorithms and techniques were established under the name of Privacy-Preserving Data Mining (PPDM) enabling data mining without breaching data privacy. The moment of actively applying these techniques can vary between the time of collection (e.g., by adding noise) and the moment of presentation (for instance reducing accuracy, suppression, and generalization) (Menders**[MEN]**).

6.6 Common Pitfalls using Production Data (LO-6.7)

Testing Analytics systems using production data sets has its pros and cons. Obvious advantages are the availability of data and its “recognizability” to the business user. Production data may show the existence of situations that “should not occur according to the design”. But, amongst others, the following disadvantages can be indicated.

Processing Time

At the stage of development and testing Functionality, it is recommended to use small data sets that provide all designed test cases – so that it can be run fast and frequently. And because analysis of errors is easier when done on a small sample. When using (full) production sets, running time may cost many hours or even days. Therefore, this approach may be harmful for a fast development and testing process.

Little Variety

The Production Set may be large (volume), but its variety may be very limited. It may not at all be useful to test processing of certain characters, error situations. A large part of the business rules may rest untouched.

Privacy Issues and/or Fines

Production Data Sets with personal identifiable information need to be prepared (by ways of anonymization/pseudonymization, see paragraph 5.5.) for testing purposes. Analyzing data sets for personally identifiable information (PII) may take up precious time. On the other hand, if anonymization/pseudonymization is poorly executed, companies run the risk of non-compliance to rules and regulations, privacy violations (for instance GDPR or local equivalent regulations) and data breaches are severely fined.

7 References

Reference	Source
[BR1]	“Testing Embedded Software”, Addison-Wesley, ISBN: 9780321159861, Broekman and Notenboom (2003).
[BT1]	“A Taxonomy for Learning, Teaching, and Assessing: A Revision of Bl’om's Taxonomy of Educational Objectives”, L. Anderson, P. W. Airasian, and D. R. Krathwohl (Allyn & Bacon 2001).
[CC1]	“Top Five Differences between DataWarehouses and Data Lakes”, C. Campbell on Blue-Granite.com (November 2015).
[DA1]	"Competing on analytics”, Harvard Business School Press, EAN: 9781633693722, T.H. Davenport and J.G. Harris (2017).
[DAM]	“The Six Primary Dimensions for Data Quality Assessment”, DAMA UK Working Group on Data Quality Dimensions, October 2013.
[DF1]	“Why The 3V’s Are Not Sufficient To Describe Big Data”, M. van Rijmenam on Datafloq.
[DG1]	“ETL Testing”, Datagaps website, https://www.datagaps.com/data-testing-concepts/etl-testing/.
[DL1]	“Data Vault Series 1 – Data Vault Overview”, D. Linstedt (2002), https://tdan.com/data-vault-series-1-data-vault-overview/5054.
[EB1]	“SmarTEST”, Egbert Bouman (2008).
[GUR]	“Star Vs Snowflake Schema: Key Differences”, on Guru99.com, https://www.guru99.com/star-snowflake-data-warehousing.html#4.
[ISO]	ISO/IEC Standard 25012:2008 on Data Quality.
[IS1]	ISTQB Glossary Testing (v3.1, 2018).
[IS2]	ISTQB CTFL Syllabus 2018 v3.1.
[JL1]	“Agile Testing, A practical guide for testers and agile teams”, L. Crispin and J. Gregory, and “More Agile Testing, Learning Journeys for the Whole Team”, L. Crispin and J. Gregory.
[KIM1]	“Fistful of flaws”, M. Ross (2003), https://www.kimballgroup.com/2003/10/fistful-of-flaws/. Link accessed 7-JUN-2020.
[KH1]	“Testing experience”, K. Haller (2014).
[LR1]	„Anchor Modeling in the data warehouse“, L. Rönnback (2007) (www.anchormodeling.com).
[MA1]	“Multi-Domain Master Data Management”, DOI, ISBN 978-0-12-800835-5, M. Allen and D. Cervo (2015)
[MEN]	“Privacy-Preserving Data Mining: Methods, Metrics and Applications”, Mendes, Ricardo & Vilela, Joao. (2017). IEEE Access. PP. 1-1. 10.1109/ACCESS.2017.2706947.
[PM1]	“Language proficiency in English”, Agarwal publication, ISBN 9789385872280, Madan Poonam (2016–2017), p138.
[RH]	“International Business Communication Standards”, Rolf Hichert, Jürgen Faisst, IBCS Version 1.1, (2017)
[SF1]	“Show me the numbers”, Stephen Few, 2nd edition, (2012).
[SG1]	“The Scrum guide”, K. Schwaber and J. Sutherland (2017).
[ST1]	“Foundation of Software Testing-ISTQB Certification”, Cengage Learning, ISBN-10: 1473764793, D. Graham, R. Black and E. van Veenendaal (2019).
[ST2]	“Testen von Data-Warehouse- und Business-Intelligence-Systemen”, Stauffer et al (2013), pages 82 and 192.
[TD1]	“Enterprise Analytics”, Thomas H. Davenport (2013), pages 12-14.
[TJ1]	”Encyclopedia of Database Systems, chapter Data Profiling”, Springer, ISBN:978-0-387-49616-0, T. Johnson (2009).
[TM1]	“TMAP Next for result driven testing”, Sogeti, ISBN: 9789072194961, Koomen, Aalst, Broekman and Vroon (2014).
[TT1]	“Test Techniques for the Test Analyst”, E. van Veenendaal (2018).