Kapitel 1

Ziele und Einsatzgebiete von Data Warehousing

»Die Möglichkeit ist die schwerste aller Kategorien.«
Søren Kierkegaard

In der heutigen IT-Landschaft sind das *Data Warehouse* (DWH) bzw. der Prozess des *Data Warehousing* feststehende Begriffe, die zum Standard der Business-Intelligence-Infrastruktur in nahezu allen größeren Unternehmen gehören. Dies überrascht kaum, gehen die Anfänge dieser Technologie doch bis in die 1980er Jahre zurück. Das von den Data-Warehouse-Koryphäen Inmon und Kimball vor über 25 Jahren maßgeblich erdachte Grundkonzept des Data Warehouse hat sich als erstaunlich robust gegenüber den mannigfaltigen Veränderungen der Informationstechnik gezeigt. Gleichwohl werden gerade in den letzten 10 Jahren die Herausforderungen für bestehende Data-Warehouse-Architekturen größer. In Zeiten von Big Data und dem unaufhaltsamen Voranschreiten der Digitalisierung kristallisieren sich zwangsläufig neue Anforderungen heraus, die in bestehende Landschaften integriert werden müssen. Auch im Hause SAP haben sich seit der Präsentation der SAP-HANA-Technologie im Jahr 2010 einige Neuerungen im Data-Warehouse-Kontext ergeben. Nachfolgend möchten wir Ihnen vor dem Hintergrund der wesentlichen Ziele und der aktuellen Anforderungen die aktuell verfügbaren Data-Warehouse-Lösungen von SAP vorstellen und Ihnen insbesondere den Ansatz des SAP-HANA-basierten *SQL Data Warehousing* näherbringen.

Definition

Die auch heute noch anerkannteste Definition eines Data Warehouse stammt aus den 1990er Jahren. Nach Bill Inmon, dem »Vater des Data-Warehousing«, ist ein Data Warehouse »*eine subjektorientierte, integrierte, nicht volatile und zeitvariante Sammlung von Daten zur Unterstützung von Managemententscheidungen*« (Inmon, 1996). Die Zielsetzung des Data Warehousing ist hier eindeutig und liegt in der Entscheidungsunterstützung von Führungspersonen und betrieblichen Fachanwender*innen. Diese Maxime gilt weiterhin und lässt sich nach dem bekannten Informatiker und Autor Ralph Kimball wie folgt gliedern (vgl. Kimball & Ross, 2013):

- **Unterstützung von Entscheidungen**

  Das Data Warehouse soll die maßgebliche Basis für eine verbesserte Entscheidungsfindung sein und mithin das zentrale Entscheidungsunterstützungs-System (auch *Decision Support System*, DSS). Dazu muss es über die entsprechenden Daten verfügen.

- **Geschäfts- und Fachorientierung**

  Das Data Warehouse soll von den Fachanwender*innen als nützlich und hilfreich erachtet werden. Technisch herausragende Lösungen sind ineffektiv, wenn sie nicht dem Geschäftsinteresse und den Bedürfnissen der Fachanwender*innen dienen.

- **Leichter Zugang zu Informationen**

  Das DWH soll die Informationen für betriebliche Fachanwender*innen und Entscheider*innen leicht zugänglich machen. Dazu gehört einerseits, dass die Systeminhalte für die Anwender*innen und nicht für die Entwickler*innen intuitiv und verständlich sind. Zum anderen müssen die Systeme und Anwendungen, die Abfragen generieren, schnell und einfach zu bedienen sein.

- **Flexibilität**

  Das DWH soll flexibel sein, da die Bedürfnisse der Anwender*innen sowie die wirtschaftlichen Rahmenbedingungen und Technologien einem stetigen Wandel unterliegen.

- **Effiziente Datenbereitstellung**

  Das DWH soll Informationen zeitgerecht bereitstellen. Die Anforderung der Echtzeit-Auswertung wird dabei immer bedeutender. Das Verhältnis zwischen Geschwindigkeit und Datenqualität ist hier zu klären.

- **Datensicherheit**

  Das DWH muss die enthaltenen Datenbestände schützen. In ihm befinden sich die wichtigsten Informationen einer Organisation. Die sichere Speicherung sowie eine adäquate Zugangsverwaltung und der Schutz vor Angriffen sind unerlässlich.

Diese Zielkategorien bilden den Grundstock zur Ausrichtung der Data-Warehousing-Prozesse. Das Zielkorsett bedarf allerdings einer weiteren Ausgestaltung im Hinblick auf konkrete Bedürfnisse, die sich gegenwärtig aus technologischer und wirtschaftlicher Perspektive ergeben und die wir im Folgenden aufbereiten wollen.

1.1 Neue Anforderungen an das Data Warehousing

4. industrielle Revolution

In den vergangenen Jahren hat der massive technologische Fortschritt nachhaltig Spuren in allen Bereichen unserer Gesellschaft hinterlassen. Der von vielen Autor*innen skizzierte Anfang einer *4. industriellen Revolution* ist im Vergleich zu bisherigen Wirtschaftsformen insbesondere durch eine explosive Dynamik gekennzeichnet, die Schnelligkeit zu einem entscheidenden wirtschaftlichen Faktor werden lässt. *Business Intelligence* sowie weitere analytische Instrumente (auch *Advanced Analytics* genannt), die nicht nur eine zeitnahe Rückschau, sondern auch zuverlässige Prognosen und Handlungsempfehlungen für die Zukunft erlauben, rücken immer mehr in den Fokus der Unternehmensführungen. Der Bedarf betrifft dabei alle Zeithorizonte, vom operativen über das taktische bis hin zum strategischen Management.

Daten als Entwicklungstreiber

Treibstoff dieser Entwicklung sind Daten. Diese werden in naher Zukunft durch die fortschreitende Digitalisierung und Etablierung von Technologien wie dem Internet der Dinge (*Internet of Things*, IoT), maschinellem Lernen (*Machine Learning*, ML) und künstlicher Intelligenz (*Artificial Intelligence*, AI) auf ein heute kaum vorstellbares Volumen ansteigen.

175 Zettabytes an Daten im Jahr 2025

Analysen des Marktforschungs- und Beratungsunternehmens International Data Corporation (IDC) und des Festplattenherstellers Seagate ergaben, dass sich die Menge an produzierten Daten im Jahr 2025 auf 175 Zettabytes belaufen wird (siehe Abbildung 1.1). Dies entspricht 175 Billionen Gigabytes und stellt eine Steigerungsrate, im Vergleich zum erwarteten Wert im Jahr 2020, von knapp 350 % dar.

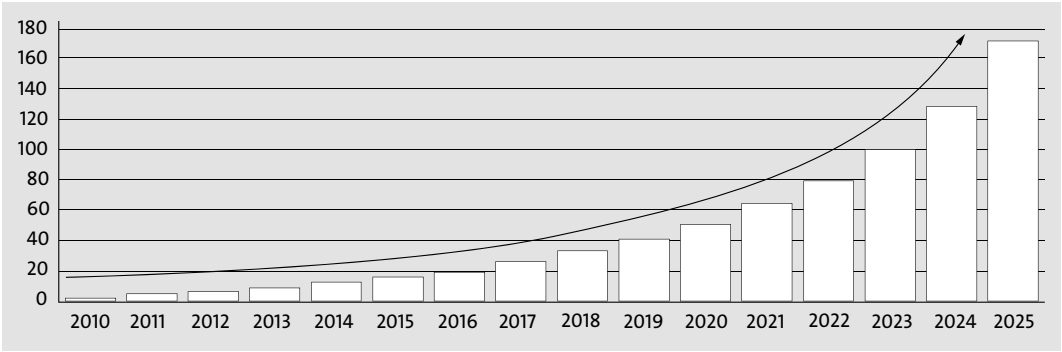

Abbildung 1.1 Prognostizierte Entwicklung der globalen Datenmenge in Zettabyte (Quelle: Reinsel et al., 2018)

Die Studie kommt darüber hinaus zu dem Schluss, dass sich – im Gegensatz zur heute ungefähren Gleichverteilung im Hinblick auf die Lagerung der

Daten bei Anwendern und Unternehmen, Mitte dieses Jahrzehnts 80 % aller Daten in den Unternehmen befinden und knapp die Hälfte aller Daten im Jahr 2025 in der Cloud liegen werden. Zudem werden Echtzeit-Daten zu diesem Zeitpunkt 30 % des Gesamtvolumens ausmachen.

Im Zusammenspiel mit den grundlegenden Zielen des Data Warehousing ergeben sich so sieben konkrete Anforderungen, die wir nachfolgend auflisten und an denen wir im Anschluss den Vergleich der verschiedenen SAP-Data-Warehousing-Ansätze ausrichten werden.

1.1.1 Charakteristika eines Enterprise Data Warehouse

EDWH-Funktionen bleiben gefragt

Das Data Warehouse ist weitgehend von der Vorstellung eines *Enterprise Data Warehouse* (EDWH) geprägt, in dem alle Daten einer Organisation zu Analysezwecken zusammengefasst werden. Die Attribute *Single Point of Truth* bzw. *Single Version of Truth* stehen für diese Idee, eine maßgebliche Dateninstanz für Analysezwecke zu schaffen. Obwohl das Konzept aufgrund der Zunahme an auswertungsrelevanten Daten in diversen Systemen in den letzten Jahren unter Druck geraten ist und verteilte DWH-Landschaften ebenfalls Vorteile bieten, sind die Kerneigenschaften eines EDWH weiterhin gefragt. Hierzu gehören Aspekte wie Datenintegration, strukturierte Datentransformationen und Datenmodellierung sowie individuelle Berechtigungskonzepte und Fragen rund um die Bereiche Dokumentation und Compliance. Weniger strukturierte *NoSQL-* und *Data-Lake-Lösungen* werden diese Funktionalität auch in Zukunft nicht bieten, sodass die klassische Ausgestaltung eines EDWH weiterhin Bestand haben wird. Dennoch wird sich die Rolle eines DWH im Unternehmenskontext verändern. Die zentralistische Ausprägung des EDWH wird an Bedeutung verlieren und sich mehr in Richtung des vom Gartner-Analysten Mark Beyer geprägten Begriffs des *logischen Data Warehouse* entwickeln, in dem der sachlogische Informationszusammenhang sowie das Augenmerk auf die virtuelle Datenintegration dominieren (vgl. Beyer, Merv, 2011).

NoSQL und Data Lakes

Der Begriff *NoSQL* umfasst Datenbanken, die nicht den klassischen relationalen Ansatz (siehe Abschnitt 5.2.1, »Relationales Datenmodell«) verfolgen und zugunsten einer besseren Skalierbarkeit auf hohe Konsistenz- und Redundanzansprüche verzichten.

Data Lakes sind Speichersysteme, in denen sowohl relational strukturierte Daten als auch NoSQL-Daten sowie Daten in Dateiformaten für Dokumente oder Bilder vorgehalten werden.

1.1.2 Flexibilität

Individuelle Schichten-architektur

Flexibilität und Anpassungsfähigkeit gehören in einer dynamischen Geschäftswelt, in der der disruptive technologische Fortschritt einen kontinuierlichen Anpassungsdruck ausübt, zu den wichtigsten Eigenschaften von Organisationen und Systemen. Für das Data Warehouse lassen sich als Bezugspunkte dieser Eigenschaften die Architektur und die Modellierung der Datenstrukturen und Datenflüsse heranziehen. Die Architektur und die Datenmodellierung korrelieren dabei stark. Einerseits ist der Aufbau der Schichtenstruktur relevant, die die Daten durchlaufen, bevor sie von spezifischen Business-Intelligence-Tools visuell aufbereitet werden. Zum anderen ist die Modellierung der Daten innerhalb der Schichten und Datenflüsse zwischen den Schichten von Bedeutung und geht eng mit der Ausgestaltung des Schichtenmodells einher. Lange waren hier die zwei Denkschulen von Bill Inmon (dreistufiger Ansatz mit Data Marts) und Ralph Kimball (zweistufiger Ansatz in ausschließlich *dimensionaler Modellierung*) vorherrschend.

Data Vault

Seit einigen Jahren ist zudem der *Data-Vault-Ansatz* von Dan Linstedt in der Praxis etabliert. (Alle drei Ansätze erläutern wir genauer in Abschnitt 3.1.3, »Architekturtypen«.) Dieser betrifft vor allem die Modellierung der Daten in der Data-Warehouse-Schicht, wirkt sich aber mittelbar auch auf die Schichtenarchitektur aus. Obwohl die Abgrenzung der einzelnen Ansätze heute nicht mehr so stark betont wird, ist generell ein Trend zur Data-Vault-Methodik erkennbar, da deren modulare Struktur vielfach den Flexibilitätsanforderungen einer stetigen Weiterentwicklung des Data Warehouse entspricht. Eine moderne Data-Warehouse-Lösung verzichtet hier jedoch im besten Falle auf Einschränkungen und erlaubt die Auswahl des konkreten architektonischen Ansatzes und der Datenmodellierung, entsprechend den vorhandenen Daten- und Analysebedürfnissen. Hinzu kommt, dass die Möglichkeit zur Standardisierung von Datenflüssen auf der Basis von *ETL*-Prozessen (Extraktion, Transformation und Laden von Daten) bzw. die weitgehende Nutzung von Datenvirtualisierungen bestehen sollte.

DWH-Architektur und -Modellierung

Weitere Informationen zur Architektur und Modellierung mit Begriffserklärungen und vielen einführenden Informationen finden Sie in Kapitel 3, »Referenzarchitektur eines modernen Data Warehouse«, und in Kapitel 5, »Methodische Grundlagen für das Data Warehousing«.

1.1.3 Agilität

Agile Entwicklungsmethoden

Ein Schlagwort, das nicht nur in der IT Hochkonjunktur hat und eng mit dem Wunsch nach organisatorischer und systemischer Flexibilität verbunden ist, ist Agilität. Seinen Ursprung haben der Begriff bzw. die methodischen Ansätze, die sich dahinter verbergen, in der Softwareentwicklung der späten 1990er Jahre. Seitdem hat der Erfolg der Vorgehensmodelle *Scrum*, *Extreme Programming* und *DevOps* ein agiles Vorgehen für nahezu alle Bereiche der Projekt- und Prozessorganisation interessant gemacht. Hierzu gehört auch das Data Warehousing, das bisher häufig mit zu langen und ungeplanten Entwicklungszeiträumen zu kämpfen hatte. Da es immer wichtiger wird, Daten und Informationen für Analysezwecke schnell zur Verfügung zu stellen, ist eine unkomplizierte Einbindung agiler Methoden ein relevantes Kriterium für die Beurteilung einer modernen Data-Warehouse-Lösung.

Kurze, iterative Entwicklungszyklen

Die Voraussetzungen zur Durchführung kurzer, iterativer Entwicklungszyklen, um zu einem möglichst frühen Zeitpunkt werthaltigen Output zu generieren, werden auch heute nicht von jeder DWH-Lösung erfüllt. Ein Blick auf die Eigenschaften, die eine spezifische Data-Warehouse-Lösung in dieser Hinsicht bereithält, ist für Anwender*innen daher von großer Bedeutung. Das zuvor genannte Kriterium der Flexibilität ist in diesem Kontext eng mit den Forderungen nach Agilität verknüpft, da sich in dieser Verbindung das Potenzial zur Reaktion auf unvorhersehbare Ereignisse offenbart.

Agile Softwareentwicklung

Weitere Informationen zu agilen Ansätzen der Softwareentwicklung sowie Scrum und DevOps finden Sie in Kapitel 4, »Entwicklungsansatz für das SAP SQL DWH«.

1.1.4 Offenheit

Steigende Systemvielfalt

Ein Data Warehouse ist per definitionem ein offenes System, da die Interoperabilität mit anderen Systemen eine wesentliche Funktion darstellt. Quellsysteme müssen angebunden werden, um die Daten für die Analyse aufzubereiten. Die konkrete Analyse erfolgt mithilfe spezifischer Visualisierungs- und Data-Science-Tools, die auf die Daten zugreifen. Die Vielfalt der an diesem Prozess beteiligten Systeme steigt im logischen DWH jedoch kontinuierlich. Die Offenheit gegenüber neuen Technologien, beispielsweise aus dem Bereich NoSQL, muss dementsprechend gewährleistet sein.

Nutzung von Tools zur agilen Entwicklung

Darüber hinaus wird die Interaktion mit weiteren Systemen relevanter. Anwendungen und Werkzeuge, die agiles Arbeiten unterstützen und die Automatisierung von Prozessschritten ermöglichen, bieten große Vorteile, und ein in dieser Hinsicht offenes DWH kann wirtschaftlicher entwickelt und betrieben werden. Eine umfassende systemische Offenheit und Konnektivität ist daher für eine zukunftsfähige Data-Warehouse-Lösung zentral.

1.1.5 Cloud

Unterschiedliche Servicemodelle

Im Thema Cloud spiegeln sich in gewisser Weise die Bedürfnisse nach einer systemischen Flexibilität und Offenheit und der hierdurch möglich werdenden agilen Arbeitsprozesse wider. Durch die Cloud wird das Data Warehouse zur Dienstleistung, wobei verschiedene Servicestufen unterschieden werden. Die einfachste Form der Cloud ist *Infrastructure as a Service* (IaaS). In diesem Szenario wird lediglich die Recheninfrastruktur, also das Hosting, als Leistung angeboten. Die Data-Warehouse-Software wird vom Kunden selbst installiert und betrieben. Beim Modell *Platform as a Service* (PaaS) erhält der Kunde darüber hinaus Möglichkeiten zur Anwendungsentwicklung durch Programmiertools und Laufzeitumgebungen. Dem Kunden obliegt in diesem Szenario lediglich die programmlogische Definition seiner DWH-Anwendung. Die weitestgehende Form der Cloud ist *Software as a Service* (SaaS). Hier sind Hardware und Software in ausdefinierter Form vorhanden, und der Kunde muss sich nur mit der Bereitstellung der Daten auseinandersetzen (siehe Abbildung 1.2).

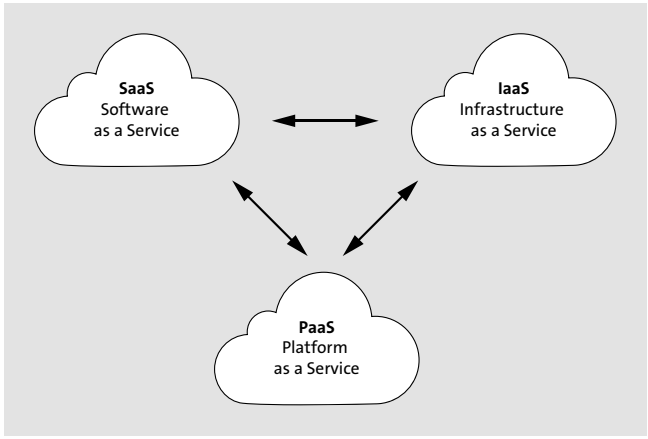

Abbildung 1.2 Cloud-Service-Varianten

Hybride DWH-Landschaften

Konzepte, speziell für die nutzerfreundlichsten Formen PaaS und SaaS, stehen für Data Warehouses noch relativ am Anfang. Die Möglichkeit zur Nut-

zung zumindest hybrider On-Premise-/Cloud-Lösungen ist jedoch schon heute ein wichtiges Beurteilungskriterium, da sich die umfassende Skalierbarkeit in der Cloud für datengetriebene Unternehmen schnell auszahlt.

1.1.6 Self-Service

Ein weiterer Aspekt, der im Data-Warehouse-Kontext in der jüngeren Vergangenheit sehr stark nachgefragt wurde und weiter an Relevanz gewinnt, ist der Bereich *Self-Service*. Der Begriff beschreibt den Umstand, dass aufgrund der voranschreitenden Digitalisierung die Fachanwender*innen in den verschiedenen funktionellen Abteilungen einer Organisation einen größeren Bedarf an individuellen Datenauswertungen für ihre spezifischen Problemstellungen haben und sich somit selbst an den Daten bedienen möchten.

Höhere IT-Kompetenz in den Fachbereichen

Immer intuitivere Visualisierungstools und Data-Science-Lösungen sowie eine höhere IT-Kompetenz der Nutzer*innen verstärken diesen Trend. Gleichwohl ist Self-Service auch heute noch für viele gewachsene Enterprise-Data-Warehouse-Lösungen kein einfaches Thema: Im klassischen Konstrukt mit Data Marts müssen die IT-Fachkräfte weiterhin die Anforderungen der Fachanwender*innen umsetzen, was Änderungsprozesse entsprechend langwierig und mühsam macht.

Das Szenario einer Schatten-IT, in der sich technologieaffine Fachanwender*innen ihre eigenen Lösungen basteln, ist immer noch gängig. Die Auflösung dieser Problematik ist daher ein zentraler Punkt moderner Data-Warehouse-Strukturen. Sie gliedert sich dabei in zwei Bereiche: Self-Service im Hinblick auf die Datennutzung durch die genannten Reporting- und Analysetools sowie Self-Service bezüglich der Datenbereitstellung, durch die es Fachanwender*innen ohne Rückgriff auf die IT möglich ist, selbst Daten in die DWH-Lösung miteinzubringen.

1.1.7 Echtzeit-Analysen

Unterstützung des operativen Geschäfts

Das Data Warehouse gehört in der historischen Entwicklung zum Bereich der Entscheidungsunterstützungs-Systeme. Diese sind grundlegend eher strategisch ausgerichtet, mit längeren Zeitfenstern, für die zyklische Auswertungen ausreichend sind. Der technologische Fortschritt hat diese Ausgangslage verändert: Daten werden zu jeder Zeit und von allen möglichen Geräten und Maschinen produziert. Trends in den sozialen Netzwerken verbreiten sich rasend schnell und haben Einfluss auf wirtschaftliche Zusammenhänge. Mit der heute zur Verfügung stehenden Rechenleistung durch

Massively Parallel Processing (MPP) und den neuen Technologien des Datenmanagements, In-Memory-Datenbanken wie SAP HANA und NoSQL-Systemen können diese Daten nahezu in Echtzeit ausgewertet werden. Diese Möglichkeit gewinnt, insbesondere für datengetriebene Geschäftsfelder, an Bedeutung. Daher sollten Sie sie bei der Betrachtung einer DWH-Lösung berücksichtigen.

Weitere Informationen zu MPP und In-Memory

Weitere Informationen zur MPP- und In-Memory-Technologie finden Sie in Abschnitt 2.2.1, »Datenbankservices«, in dem wir die entsprechenden Funktionen von SAP HANA erläutern.

1.2 Data-Warehousing-Ansätze von SAP im Vergleich

Drei komplementäre SAP-DWH-Ansätze

Nachdem wir Ihnen die Ziele und Anforderungen des modernen Data Warehousing erläutert haben, widmen wir uns im Folgenden überblicksartig den entsprechenden Ansätzen von SAP. Abbildung 1.3 zeigt die derzeit unterstützten Data-Warehouse-Ansätze SAP BW/4HANA, SAP SQL Data Warehousing und SAP Data Warehouse Cloud.

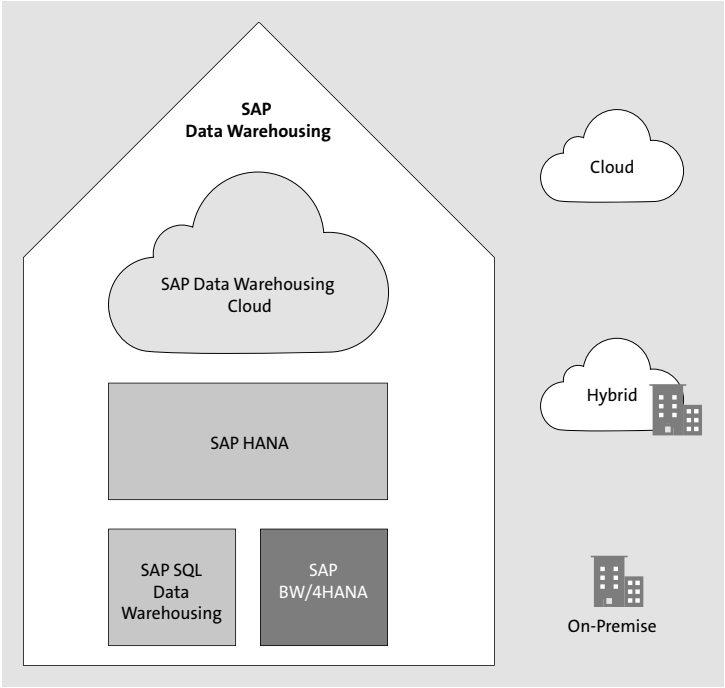

Abbildung 1.3 Die verschiedenen SAP-Data-Warehousing-Ansätze (Quelle: SAP)

Die letzte Lösung ist, wie der Name schon sagt, nur in der Cloud verfügbar. Die anderen beiden Lösungen stehen als On-Premise-Versionen zur Verfügung, wobei das SAP SQL DWH durch SAP HANA Cloud und SAP HANA beide Optionen bietet. In den folgenden Abschnitten arbeiten wir die jeweiligen Stärken und Unterschiede vor dem Hintergrund der vorgestellten Anforderungen heraus, um ein vollständiges Bild des Einsatzgebietes der jeweiligen Lösung zu zeichnen. Grundsätzlich möchten wir betonen, dass sich die Ansätze komplementieren und Mischformen und Hybrid-Architekturen übergreifende Nutzenvorteile versprechen.

1.2.1 SAP BW und SAP BW/4HANA

Das *SAP Business Warehouse* (SAP BW) ist eine Konstante im SAP Data Warehousing. Die Software wurde 1997 veröffentlicht und existiert aktuell in Version 7.5. Seit Version 7.3 vertreibt SAP die Lösung *powered by SAP HANA* und integriert die alte ABAP-Welt mit der neuen SQL-Ausrichtung der SAP-HANA-Plattform. Im Jahr 2016 stellte SAP daneben die Neuentwicklung SAP BW/4HANA vor (siehe Abbildung 1.4). Dabei handelt es sich zwar nicht um den offiziellen Nachfolger von SAP BW. Dennoch hält SAP BW/4HANA einen ähnlichen Funktionsumfang bereit, und die wesentlichen Architekturkonzeptionen von SAP BW haben weiterhin Bestand.

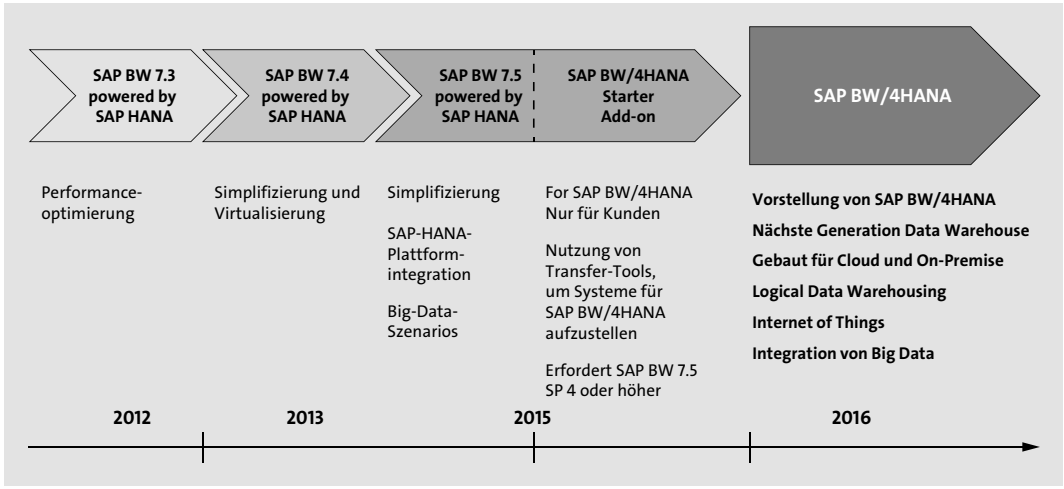

Abbildung 1.4 Evolution von SAP BW und SAP BW/4HANA (Quelle: SAP)

SAP BW/4HANA

Da SAP BW/4HANA nur die Nutzung von HANA-optimierten Objekten unterstützt und im Gegensatz zu SAP BW nicht auf anderen Datenbanken eingesetzt werden kann, werden in absehbarer Zukunft beide Ansätze parallel nebeneinander existieren. Langfristig wird SAP BW jedoch eingestellt und

SAP BW/4HANA als logischer Nachfolger fungieren. SAP skizziert den Weg von SAP BW 7.X zu SAP BW/4HANA eindrücklich und hat angekündigt, dass Innovationen ausschließlich in SAP BW/4HANA einfließen werden (siehe *http://s-prs.de/v781700*). Darüber hinaus ist SAP BW noch Teil des SAP-NetWeaver-Angebots, dessen Support in den nächsten Jahren sukzessive ausläuft (siehe SAP-Hinweis 1648480). Daher konzentrieren wir uns in unseren folgenden Ausführungen auf SAP BW/4HANA und beschreiben diesen Ansatz, der jedoch maßgeblich durch die Historie von SAP BW geprägt ist.

Anwendungsorientierter Ansatz

Plug-&-Play-BI-Lösung

Durch die technische Nähe zum klassischen SAP-BW-Ansatz, der immer darauf ausgerichtet war, eine, das SAP-ERP-System ergänzende, komplett konfektionierte Plug-&-Play-BI-Lösung zu sein, gilt SAP BW/4HANA auch als anwendungsorientierter Data-Warehousing-Ansatz. Das Grundziel der Entwicklung lag vor allem darin, IT-Landschaften mit SAP-basierten Datenquellen bzw. einer SAP-BW-Historie für die neuen Herausforderungen des Data Warehousing zu rüsten. Zentrales Element auf diesem Weg ist die SAP-HANA-Plattform, deren Stärken wir Ihnen in Kapitel 2, »Einführung in SAP HANA als Plattform«, vorstellen, da sie global für alle SAP-Data-Warehousing-Ansätze gelten. Im Hinblick auf die in Abschnitt 1.1, »Neue Anforderungen an das Data Warehousing«, aufgeführten zentralen Anforderungen an das zukünftige Data Warehouse lassen sich für SAP BW/4HANA verschiedene Aspekte festhalten, die wir Ihnen im Folgenden vorstellen.

Enterprise Data Warehouse

Content der BW-Evolution

SAP BW/4HANA bringt aufgrund dessen Historie alle Eigenschaften eines klassischen EDWH mit. Sie können unterschiedliche Datenquellen anbinden, wobei die Konnektivität zu anderen SAP-Systemen durch *Operational Data Provisioning* (ODP) und *Operational Delta Queue* (ODQ) hervorzuheben ist. Das ABAP-Know-how aus der BW-Evolution kommt hier voll zum Tragen und ermöglicht einfache und schnelle BI-Lösungen. Auch die strukturierte Aufbereitung der Daten durch Transformationen im Rahmen von ETL-Strecken sowie virtuelle Datenverknüpfungen sind im Funktionsumfang enthalten. Darüber hinaus verfügt SAP BW/4HANA über dezidierte Berechtigungskonzepte und wird höchsten Ansprüchen in puncto Dokumentation und Compliance gerecht. Es eignet sich daher in einer zunehmend verteilten DWH-Landschaft dazu, die immer noch benötigten Aufgaben einer umfassenden Data-Warehouse-Lösung wahrzunehmen und dabei ein wichtiger Teil des sachlogischen Gesamtzusammenhangs in Form des logischen Data Warehouse zu sein.

Weitere Informationen zu ODP und ODQ

Weitere Informationen zu ODP und ODQ finden Sie im SAP Help Portal unter *http://s-prs.de/v781701*.

Flexibilität

Vereinfachte Datenarchitektur durch LSA++

SAP BW verfügt mit der *Layered Scalable Architechture* (LSA) bereits seit etlichen Jahren über ein spezifisches Schichtenmodell, das die Daten durchlaufen, bevor sie von BI- und Reporting-Tools konsumiert werden können. Mit der Adaption der SAP-HANA-Datenbank wurde diese Architektur angepasst und zu *LSA++* weiterentwickelt. Dabei stand vor allem eine Verschlankung der bisherigen LSA im Vordergrund. So wurde die Anzahl der Modellierungsobjekte und zwingend benötigten Schichten reduziert, die Objekte für die SAP-HANA-Datenbank optimiert und klassisches ETL sowie Datenvirtualisierungen ermöglicht. Insgesamt wurde auf diese Weise die Flexibilität erhöht, und es besteht ein Spielraum für die konkrete Ausgestaltung der LSA++, wenngleich im Unterschied zum nativen SAP SQL Data Warehousing keine völlige Freiheit hinsichtlich der Schichten und Datenmodellierung besteht.

Agilität

Agile Methoden sind möglich

Agile Methoden der Softwareentwicklung setzen auf kurze, iterative und inkrementelle Auslieferungszyklen, die auf eine frühzeitige Nutzen- und Wertstiftung für die Anwender*innen ausgerichtet sind. Die Möglichkeit zur agilen Entwicklung wird im Bereich des Data Warehousing stark von der Flexibilität der Architektur bestimmt. Die Konzeptionierung der LSA++ hat insofern dazu beigetragen, dass SAP BW/4HANA viel agiler entwickelt werden kann als ältere SAP-BW-Systeme. So werden beispielsweise durch eine feldbasierte Modellierung inkrementelle und iterative Vorgehensweisen möglich und die Geschwindigkeit bis zur Realisierung nutzbarer Änderungen erhöht. Dennoch sind aktuell nicht alle aus der Softwareentwicklung bekannten agilen Entwicklungsansätze im Rahmen von SAP BW/4HANA anwendbar. Für parallele Entwicklungsarbeiten und Praktiken wie DevOps fehlt es bisher an einer geeigneten Versionsverwaltung für den Quellcode, die die Integration von unterschiedlichen Entwicklungsartefakten einfach ermöglicht. In dieser Hinsicht ist die Agilität von SAP BW/4HANA nicht so hoch wie die des SAP SQL Data Warehouse.

Offenheit

Offenheit durch SAP HANA

SAP BW/4HANA erfüllt die Anforderungen an ein offenes Data Warehouse nur in eingeschränkter Form. Eine umfassende Offenheit gegenüber unterschiedlichen Quellsystemen und Drittanbieter-Software zur Datenvisualisierung ergibt sich eher in Hybrid-Szenarien, in denen die SQL-Konnektivität der SAP-HANA-Plattform und damit Kernfunktionen des SAP SQL Data Warehousing genutzt werden. So sind Datenquellen aus dem Bereich No-SQL, allen voran das Framework Apache Hadoop, über die SAP-HANA-nativen Paketlösungen für das Enterprise Information Management (EIM) oder *SAP Vora* anbindbar. Visualisierungstools von Drittanbietern können Inhalte performant über den SAP-HANA-Client aus SAP BW/4HANA konsumieren, was jedoch eine aufwendigere Hybrid-Architektur erfordert. Im Hinblick auf die Einbindung von Werkzeugen zur Unterstützung von agilen Entwicklungsansätzen, wie beispielsweise externen Versionskontrollsystemen (z. B. Git) oder Automatisierungstools (z. B. Jenkins), bestehen ebenfalls keine geeigneten Schnittstellen.

Insgesamt ist der Funktionsumfang von SAP BW/4HANA damit in puncto Offenheit gegenüber früheren SAP-BW-Versionen zwar wesentlich verbessert, im Vergleich zum SAP SQL Data Warehousing ist SAP BW/4HANA allerdings immer noch überwiegend auf den SAP-Kosmos ausgerichtet, und der native SAP-HANA-Ansatz bietet hier eine größere Freiheit, die gleichwohl in einer komplexeren Hybrid-Struktur auch für SAP BW/4HANA genutzt werden kann.

Cloud

Hybrid-Lösungen mit SAP HANA Cloud

SAP BW/4HANA ist lediglich als On-Premise-Version verfügbar. Als Cloud-Optionen bieten sich daher nur das simple Hosting durch einen Dienstleister wie Amazon Web Services (AWS), bei dem im Sinne von IaaS lediglich die fremde Infrastruktur genutzt wird, sowie das Zusammenspiel mit SAP HANA Cloud und SAP Data Warehouse Cloud als SaaS an. Strategisch ergeben sich für SAP BW/4HANA damit nur Hybrid-Szenarien, um in den Genuss von umfassenden Cloud-Vorteilen, wie flexibler Skalierbarkeit, zu kommen.

Self-Service

Im Hinblick auf das Stichwort Self-Service können die Fachbereiche vor allem über die SAP-Frontend-Tools SAP Lumira und SAP Analytics Cloud individuelle und interaktive Reports auf der Basis von SAP BW/4HANA erstellen. Der Rückgriff auf Software von Drittanbietern, wie Tableau, Qlik oder Microsoft Power BI, erfordert in performanter Weise den Umweg über eine Hybrid-Architektur mit dem SAP SQL Data Warehousing. Hierbei werden die Daten direkt über die SAP-HANA-Datenbank zur Verfügung gestellt und von den Visualierungstools über einen SAP-HANA-Client (ODBC) konsumiert. Einzelheiten zu den Schnittstellen von SAP HANA finden Sie in Ab-

schnitt 2.1.1, »Serverstruktur«. Darüber hinaus bleibt die Möglichkeit eines Zusammenspiels mit SAP Data Warehouse Cloud, das in dieser Hinsicht über große Stärken verfügt. Mehr dazu erfahren Sie in Abschnitt 1.2.2, »SAP Data Warehouse Cloud«.

Echtzeit-Analysen

Echtzeit durch SAP-HANA-Performance

In Bezug auf die heute vielfach geforderte Auswertung von Daten in Echtzeit bringt SAP BW/4HANA alle nötigen Voraussetzungen mit. Durch die In-Memory-Verarbeitung und die weiteren performancefördernden Eigenschaften der SAP-HANA-Datenbank sowie entsprechende Schnittstellen, die Datenvirtualisierungen, Datenstreaming und Delta-Replikationen ermöglichen, besteht hier eine solide Grundlage. Auf der Basis von LSA++ können Sie die Architektur von SAP BW/4HANA zudem so gestalten, dass die Bedürfnisse nach Datenauswertungen in Echtzeit umfassend befriedigt werden können.

Einsatzgebiete

Stärken im SAP-Umfeld

Die Prüfung anhand der Kriterien eines modernen Data Warehouse zeigt, dass SAP BW/4HANA den Funktionsumfang eines klassischen Enterprise Data Warehouse bereithält und damit Teilaufgaben im Gesamtkontext eines logischen Data Warehouse übernehmen kann. Durch die Adaption der SAP-HANA-Technologie hat das Produkt nicht nur an Leistungsstärke, sondern auch an Flexibilität und Offenheit gewonnen. Die Möglichkeiten, die Entwicklung von SAP BW/4HANA agil voranzutreiben und auf die sich schneller ändernden Umweltbedingungen reagieren zu können, haben sich im Vergleich zu den bisherigen SAP-BW-Formen verbessert. Die zentrale Integration großer Datenmengen zu Auswertungswecken ist mit SAP BW/4HANA möglich. Massendaten aus IoT und den sozialen Medien sowie sonstige Kategorien unstrukturierter Datenmengen werden zwar nicht unmittelbar in SAP BW/4HANA verarbeitet, durch die Anbindung entsprechender Speziallösungen können aber auch solche Daten Eingang in den strukturierten Enterprise-Data-Warehousing-Prozess finden. Seine spezifischen Stärken spielt SAP BW/4HANA dabei weiter in Systemlandschaften mit SAP-Einschlag aus. Die historisch gewachsenen Reporting-Lösungen für SAP ERP oder sonstige SAP-Domänen, sind hier weiterhin konfektioniert zu bekommen. Letztlich bieten sich insbesondere dann Vorteile, wenn es darum geht, bestehende ABAP-Expertise im Unternehmen weiterhin zu nutzen.

1.2.2 SAP Data Warehouse Cloud

1. SaaS-DWH-Lösung

Im Mai 2019 hat SAP auf der jährlichen SAPPHIRE-NOW-Konferenz mit *SAP Data Warehouse Cloud* eine rein cloudbasierte Data-Warehouse-Lösung präsentiert. SAP Data Warehouse Cloud ist Teil der SAP-HANA-Cloud-Services, die das Ziel verfolgen, ein vollumfängliches Datenmanagement zur Entscheidungsunterstützung auf Basis der SAP-HANA-Technologie als SaaS zu offerieren. Nach einer Beta-Testphase ist SAP Data Warehouse Cloud seit Ende des Jahres 2019 verfügbar, und im Jahr 2020 wurden bereits weitere Funktionen entwickelt. Die strategische Ausrichtung von SAP zeigt damit entsprechend dem Slogan »Cloud first« deutlich in Richtung cloudbasierter oder kurzfristig zumindest hybrider Data-Warehousing-Strukturen.

Fachanwendungsorientierter Ansatz

Zukünftig vollwertige DWH-Lösung

Mit SAP Data Warehouse Cloud bietet SAP einen flexiblen Cloud-Service, mit dem Unternehmen ihre bestehende Data-Warehouse-Lösung zunächst einmal einfach erweitern können, um hybride Strukturen zu schaffen und sich nach ihrem Ermessen langsam in Richtung Cloud zu orientieren. Im Vergleich zu den übrigen Lösungen SAP BW/4HANA und SAP SQL Data Warehousing stehen vorerst die Fachanwender*innen als Zielgruppe im Fokus. Diese sollen in einer ihnen verständlichen Weise in die Lage versetzt werden, ohne großen Verwaltungsaufwand und vertieftes IT-Wissen entsprechend ihren fachlichen Bedürfnissen Daten auszuwerten, um ihre Entscheidungsfindung zu erleichtern. Der Self-Service-Gedanke spielt hier eine wesentliche Rolle. Dies ist allerdings nur als Ausgangssituation zu bewerten: Auf Basis der Weiterentwicklungen im Jahr 2020 ist deutlich zu erkennen, dass SAP mit Data Warehouse Cloud mittelfristig ein vollständiges DWH als SaaS in der Cloud etablieren möchte. In Bezug auf die Kriterien eines modernen Data Warehouse lassen sich daher verschiedene Aspekte aufführen, die wir Ihnen im Folgenden vorstellen.

Enterprise Data Warehouse

Wertvolle Hybrid-Option

Gegenwärtig ist SAP Data Warehouse Cloud als Erweiterung für Data Warehouses gedacht, die bereits als On-Premise-Version oder als IaaS- bzw. PaaS-Cloud-Variante betrieben werden. SAP plant den Funktionsumfang jedoch auszubauen, sodass zukünftig auch die Rolle eines klassischen EDWH für die SaaS-Lösung denkbar ist. Momentan werden Systeme wie SAP BW/4HANA oder SAP SQL DWH als vorgeschaltete Instanz noch benötigt, um die Kerneigenschaften eines EDWH umfassend abdecken zu können. SAP SQL DWH bietet hier im Vergleich zu SAP BW/4HANA den Vorteil, dass es nicht nur on-premise, sondern auch in der Cloud betrieben werden kann.

In Kombination mit SAP Data Warehouse Cloud können Sie so bereits heute ein rein cloudbasiertes EDWH entwickeln (vgl. Meier, 2020a, 2020b, 2020c, 2020d), sodass SAP Data Warehouse Cloud als eigenständige EDWH-Lösung nicht unbedingt benötigt wird. In mehr und mehr verteilten Data-Warehouse-Landschaften kann SAP Data Warehouse Cloud auf diese Weise durch die spezifische Ausgestaltung für betriebliche Fachbereiche schon jetzt einen Beitrag leisten und für eine deutliche Erhöhung des Self-Service im logischen Data Warehouse sorgen. Eine umfassende Möglichkeit zur Überführung von Modellen des SAP SQL Data Warehousing in SAP Data Warehouse Cloud ist dabei in absehbarer Zeit ein realistisches Szenario.

Flexibilität

Simple Architektur für Fachbereiche und IT

SAP Data Warehouse Cloud liefert aktuell vor allem einfache Werkzeuge zur Datenmodellierung, die auf den Bedarf von Fachanwender*innen zugeschnitten sind. Für komplexe Datenstrukturen bieten die beiden übrigen SAP-Data-Warehousing-Ansätze zurzeit noch qualifiziertere Mittel, um auf die Architektur der Datenströme einzuwirken. Vorgelagerte DWH-Lösungen bieten zudem den Vorteil, dass weniger Daten persistent in die Cloud repliziert werden. Hybrid-Strategien, die darauf ausgerichtet sind, bestehendes BW- oder SQL-Know-how zu ergänzen, sind daher in naher Zukunft ein pragmatisches und logisches Mittel, um den Flexibilitätsansprüchen zu genügen. Da die SAP-HANA-Datenbank auch das zentrale Moment der SAP Data Warehouse Cloud bildet, ergeben sich in diesem Zusammenhang Vorteile für Anwender*innen des nativen SAP SQL Data Warehousing. Sie können beispielsweise vorhandene Modelle ohne weiteres in SAP Data Warehouse Cloud umsetzen. Auf diesem Weg kann ein sukzessiver Umbau der On-Premise-Struktur in die Cloud erfolgen. Auf lange Sicht ist SAP Data Warehouse Cloud als vollwertige Cloud-Lösung zu sehen, in der alle Funktionalitäten eines Data Warehouse als SaaS zur Verfügung stehen.

Agilität

Agilität durch Self-Service- und SQL-Ansatz

Hinsichtlich agiler Entwicklungsmethoden ist bei SAP Data Warehouse Cloud aufgrund des derzeitigen Fokus auf die Fachanwender*innen vor allem auf deren Agilität abzustellen. Da die Fachbereiche selbst auf die benötigten Daten zugreifen und den Weg bis zur Visualisierung der Daten in interaktiven Berichten und Dashboards vollziehen können, ohne Unterstützung von der IT zu benötigen, ergibt sich hier eine hohe Agilität. Stellenübergreifende Abhängigkeiten werden auf diese Weise abgebaut und die Zeit zur Erstellung nutzen- und wertstiftender Ergebnisse verkürzt. Hierbei helfen zahlreiche Vorlagen, die Best-Practice-Szenarien für übliche Geschäftsprozesse repräsentieren. In Bezug auf komplexe Datenstrukturen

ergibt sich ein geteiltes Bild: Soll eine solche Aufbereitung weiter on-premise erfolgen, kommt es auf das agile Potenzial der entsprechenden Data-Warehouse-Lösung an. Soll die Aufbereitung ebenfalls in SAP Data Warehouse Cloud erfolgen, ergibt sich ein agiles Bild, das im Wesentlichen den Möglichkeiten des nativen SAP SQL Data Warehousing entspricht (siehe Abschnitt 1.2.3, »SAP SQL Data Warehousing«).

Offenheit

Offenheit der SAP-HANA-Plattform

SAP Data Warehouse Cloud stellt eine Data-Warehouse-Lösung dar, die höchsten Ansprüchen an eine systemische Offenheit genügt. Durch die SQL-Konnektivität der SAP-HANA-Datenbank ergibt sich sowohl auf Dateneingangs- als auch auf Datenausgangsseite eine hohe Flexibilität. Unterschiedliche Quellsysteme können angebunden und nicht nur durch das integrierte Visualisierungstool SAP Analytics Cloud, sondern auch durch Drittanbieter-Frontends wie Tableau, Qlik oder Microsoft Power BI ausgewertet werden. Darüber hinaus könnte in Zukunft auch die Einbindungsmöglichkeit von Tools zur Unterstützung von agilen Entwicklungsansätzen wie Versionskontrollsystemen und Automatisierungstools gegeben sein, die den umfassenden Möglichkeiten des SAP SQL Data Warehousing entsprechen (siehe Abschnitt 1.2.3, »SAP SQL Data Warehousing«).

Cloud

Volle SaaS-Flexibilität

In Bezug auf das Kriterium Cloud setzt SAP Data Warehouse Cloud als erste reine SaaS-Data-Warehouse-Lösung neue Maßstäbe. Anwender*innen kommen zukünftig ohne größeren administrativen Aufwand und Gedanken über die Hardware schnell in den Genuss einer vollumfänglichen Data-Warehouse-Lösung. Die Skalierbarkeit entsprechend den spezifischen Bedürfnissen, der geringe Instandhaltungsaufwand und eine insgesamt erhöhte Kostentransparenz sind Argumente für SAP Data Warehouse Cloud.

Self-Service

Spaces für Fachanwender*innen

Wie bereits angedeutet, liegt ein Schwerpunkt von SAP Data Warehouse Cloud auf dem Aspekt Self-Service-BI (BI = Business Intelligence). Zur Unterstützung dieses Gedankens setzt SAP Data Warehouse Cloud auf ein Konzept von *Spaces*, das eine isolierte Umgebung für Geschäftsbereiche oder spezifische Anwendungsfälle von einzelnen Personen oder Teams zur Verfügung stellt. Weitere Informationen zu Spaces finden Sie in Kapitel 13, »SAP Data Warehouse Cloud«. In den Spaces können die Fachanwender*innen ihre Daten frei auswählen und bereitstellen sowie nach ihren Bedürfnissen und ihrem Verständnis modellieren und auswerten. Da die Datenbankobjekte im jeweiligen Space isoliert werden, besteht hier völlige Freiheit und keine Gefahr, durch veränderte Objekte andere DWH-Prozesse im

Unternehmen negativ zu beeinflussen. Die einzige Abhängigkeit zur IT besteht in der Bereitstellung der Spaces und etwaigen weiteren Berechtigungen. Ein weiterer Vorteil ist die unmittelbare Anbindung des Frontend-Tools SAP Analytics Cloud als Teil der SAP-HANA-Cloud-Services. Die Fachanwender*innen können so in einem User Interface den gesamten Prozess der Datenaufbereitung und -visualisierung vollziehen und verfügen somit über eine Ende-zu-Ende-Lösung. Die Nutzung anderer Frontend-Tools sowie weiterer Data-Science-Werkzeuge steht den Anwender*innen gleichwohl offen.

Echtzeit-Analysen

Aufgrund der Möglichkeiten, die die SAP-HANA-Plattform durch die hohe Performance der In-Memory-Technologie und die Verarbeitung von Daten durch Virtualisierungen, Streams und Delta-Replikationen bieten, sind auch in SAP Data Warehouse Cloud Echtzeit-Lösungen möglich. Im Vergleich zu den On-Premise-Lösungen besteht insoweit ein Vorteil, dass mit den sogenannten Open Connectors der Cloud-Umgebung eine Vielzahl an Konnektoren zu weiteren Cloud-Diensten zur Verfügung stehen. Gepaart mit Agilität und Flexibilität hinsichtlich der Modellierung und Auswertung der Daten in den Spaces können Sie relevante Ergebnisse nicht nur nahezu in Echtzeit abrufen, sondern auch sehr schnell erarbeiten.

Einsatzgebiete

Hybrid-Szenarien mit erhöhtem Self-Service

SAP Data Warehouse Cloud befindet sich aktuell noch in der Einführungsphase. Nichtsdestotrotz stellt das Produkt für Kunden, die bereits SAP BW oder HANA-basierte Produkte einsetzen, eine interessante Erweiterung zur Unterstützung von Fachabteilungen dar. Prädestiniert ist die Lösung für die Aufgabe, konsolidierte Daten einer etablierten Data-Warehouse-Landschaft mit Informationen speziell für einen Fachbereich in Beziehung zu setzen. Die Fachanwender*innen haben hier freie Hand und können Daten, z. B. aus Microsoft Excel oder aus Online-Quellen, leicht integrieren.

Strategischer Wechsel in die Cloud

Darüber hinaus ist SAP Data Warehouse Cloud für praktisch jeglichen Anwendungsfall eine strategische Cloud-Option. Durch die perspektivische Ausgestaltung als vollumfängliche Data-Warehouse-SaaS-Lösung, die sowohl die Anforderungen von erfahrenen Data-Warehouse-Entwickler*innen und IT-Fachleuten als auch die Bedürfnisse der Fachanwender*innen befriedigt, würden zukünftig alle Stakeholder mitgenommen. Die schnelle Verfügbarkeit in der Cloud, die einheitliche Benutzeroberfläche über den gesamten Prozess des Data Warehousing sowie die Spaces mit einer Reihe an Vorlagen machen mittelfristig nicht nur hybride Szenarien für datengetriebene Unternehmen attraktiv, sondern werben für eine stetige Verlagerung des Data Warehousing in die Cloud. Die Systemlandschaft kann dabei weiterhin sehr heterogen aussehen, jedoch mit dem Vorteil, dass analytische Schattensysteme und Excel-Durcheinander an Attraktivität verlieren, was die Datenkontrolle, -transparenz und -konsistenz insgesamt erhöht.

1.2.3 SAP SQL Data Warehousing

SQL Data Warehousing auf der Basis von SAP HANA

Seit der Vorstellung von SAP HANA im Jahr 2010 gibt es aufgrund der im Vergleich zu üblichen relationalen Datenbank-Managementsystemen bestehenden Performance-Vorteilen der In-Memory-Technologie in der Praxis Überlegungen, SAP HANA nativ als Data Warehouse zu verwenden. Mit der Weiterentwicklung der SAP-HANA-Datenbank zu einer Plattform haben sich diese Überlegungen weiter verfestigt – insbesondere durch die neuen Möglichkeiten von SAP HANA 2.0, das 2017 veröffentlicht wurde. Damit einher ging die Erneuerung der Entwicklungsplattform *SAP HANA Extended Application Services, Advanced Model* (SAP HANA XSA) und des integrierten Anwendungsservers zur offenen Kommunikation des nativen SAP SQL Data Warehousing bzw. SAP SQL Data Warehouse. Dieser Ansatz steht seitdem neben dem applikationsorientierten SAP-BW/4HANA-System und der Cloud-Lösung SAP Data Warehouse Cloud und kann eigenständig oder in hybrider Form mit den beiden übrigen DWH von SAP betrieben werden.

SQL-Ansatz Best-of-Breed

SQL-basierte Werkzeuge

Da das Herzstück der SAP-HANA-Plattform in der SAP-HANA-Datenbank besteht, deren Abfragesprache die *Structured Query Language* (SQL) ist, wird die native Verwendung der SAP-HANA-Plattform zum Data Warehousing auch als *SQL-Ansatz* bezeichnet. Durch den universellen Gebrauch der Standardsprache SQL im Bereich der relationalen Datenbank-Managementsysteme zeichnet sich die Vorgehensweise vor allem durch ein hohes Maß an systemischer Offenheit und Entwicklungsagilität aus. Hierdurch können Sie nicht nur das in der SAP-HANA-Plattform enthaltene Toolset, sondern auch viele weitere geläufige Werkzeuge der Software- und Datenbankentwicklung flexibel nutzen. Für jeden Einzelfall kann so eine optimale Zusammenstellung verschiedener SAP- und Nicht-SAP-Komponenten erfolgen. Diese Vorgehensweise wird auch als *Best-of-Breed* bezeichnet. SAP SQL Data Warehouse bietet sich so auch für komplexe Szenarien in unterschiedlichen DWH-Landschaften an. Solide SQL-Kenntnisse sind dabei hilfreich, da sie die Basis der zum Einsatz kommenden Werkzeuge bilden. Bezüglich der Anforderungen an ein modernes DWH sollten Sie darüber

hinaus noch einige weitere Aspekte berücksichtigen, die wir Ihnen im Folgenden vorstellen.

Enterprise Data Warehouse

Das SAP SQL Data Warehouse bringt alle Voraussetzungen zur Gestaltung eines EDWH mit. Sie können Datenquellen umfassend anbinden und die Daten per ETL oder virtuell ins DWH laden. Berechtigungen können individuell vergeben werden, und durch die sehr freie Form der Datenmodellierungen können Sie auch Compliance- und Dokumentationsanforderungen gut integrieren. SAP SQL DWH ist somit ebenso wie SAP BW/4HANA in der Lage, eine tragende Rolle im Kontext eines logischen Data Warehouse zu übernehmen. Im Vergleich zum applikationsorientierten SAP-BW/4HANA-System gleicht das SAP SQL Data Warehouse dabei mehr einem Bausatz als einer konfektionierten Lösung. Vorteile ergeben sich insbesondere im Hinblick auf Lösungsansätze, die über den Standardrahmen von SAP BW/4HANA hinausgehen. In diesen Szenarien können Sie mit SAP SQL Data Warehousing anpassungsfähig und agil agieren.

Flexibilität

Freie Architektur und Datenmodellierung

SAP SQL Data Warehouse erlaubt Ihnen eine freie Gestaltung der Schichtenarchitektur und der Datenmodellierung. Sei es die Anzahl der Schichten, die Nutzung von Data Marts, die Modellierung in *dritter Normalform* (3NF), dimensional oder nach der Data-Vault-Methodik – die architektonische Ausgestaltung kann entsprechend den individuellen Gegebenheiten erfolgen. Dabei sind sowohl das klassische ETL als auch Datenvirtualisierungen möglich. Die Flexibilität ist in diesem Bereich besonders hoch und fördert die Möglichkeiten einer kontinuierlichen Anpassung durch agile Entwicklungsmethoden sowie eine Erhöhung des Automatisierungsgrades bei der Weiterentwicklung des SAP SQL DWH.

Weitere Infos zu Architektur und Datenmodellierung

Aspekte zu DWH-Architektur und zur Datenmodellierung erläutern wir allgemein in Kapitel 3, »Referenzarchitektur eines modernen Data Warehouse«, und in Kapitel 5, »Methodische Grundlagen für das Data Warehousing«.

Agilität

Paralleles Arbeiten in Sandboxes

Durch die freie Architektur und Datenmodellierung im SAP SQL Data Warehouse sowie durch die erleichterte Verwendung von Drittanbieter-Software und -Tools ist der SQL-Ansatz prädestiniert für agile Entwicklungsansätze.

SAP HANA 2.0 bringt zudem Mittel und Werkzeuge mit, die im Vergleich zur bisherigen Data-Warehouse-Entwicklung im SAP-Umfeld neue Maßstäbe setzen. So arbeiten Entwickler*innen auf dem in der SAP-HANA-Plattform integrierten SAP-HANA-XSA-Anwendungsserver in isolierten Arbeitsbereichen oder *Sandboxes*, die das parallele Entwickeln enorm erleichtern. Mitverantwortlich für diese Erleichterung ist eine zweite technische Neuerung. Die zentrale Entwicklungsumgebung *SAP Web IDE* erlaubt die Speicherung des Quellcodes im verteilten Versionskontrollsystem Git. Hierdurch verbessert sich die Versionsverwaltung, und das Zusammenfügen separat erarbeiteter Entwicklungen gelingt reibungsloser.

DevOps-Implementierung

Insgesamt werden so kurze, iterative und inkrementelle Auslieferungszyklen durch eine kontinuierliche Integration (*Continiuous Integration*) und eine kontinuierliche Auslieferung (*Continuous Delivery*) im Sinne der *DevOps-Philosophie* möglich. Dies bedeutet einen erheblichen prozessualen Mehrwert für die Erstellung komplexer Data-Warehouse-Strukturen und ist ein großer Vorteil von SAP SQL Data Warehousing.

Offenheit

Umfassende Offenheit durch SQL

In puncto Offenheit bietet SAP SQL Data Warehousing aufgrund der universellen Möglichkeiten des SQL-Standards und der Ausgestaltung als Plattform auf der Basis des Anwendungsservers SAP HANA XSA im Vergleich der drei SAP-Data-Warehousing-Ansätze aktuell das größte Potenzial. So ist nicht nur bezüglich der Quellsysteme und Datenvirtualisierungstools die volle Flexibilität gewährleistet, sondern im Rahmen der für SAP SQL Data Warehousing empfohlenen DevOps-Philosophie können Sie auch die aus der Softwareentwicklung bekannten Tools zur Unterstützung und Automatisierung der Auslieferungs-Pipeline wie Git, Jenkins und ähnlichen Tools nutzen. Darüber hinaus haben Sie in der Entwicklungsumgebung SAP Web IDE die Option, Applikationen *full-stack* zu entwickeln und sich auf diese Weise weitere Optionen des Datenmanagements zu schaffen. (Mehr zu diesem Thema erfahren Sie in Abschnitt 2.2.4, »Anwendungsentwicklung«.) Beachten Sie, dass durch die strukturelle Konformität der On-Premise-Version mit dem SAP-HANA-XSA-Anwendungsserver und der Cloud-Version SAP HANA Cloud auf Basis des Cloud-Standards *Cloud Foundry* eine Auslieferung von Entwicklungen in die jeweils andere Umgebung unmittelbar möglich ist. Diese strukturelle Konformität der SAP-HANA-Plattform, on-premise und in der Cloud, führt darüber hinaus dazu, dass die Methodik des SAP SQL Data Warehousing auch für SAP Data Warehouse Cloud adaptiert werden kann.

Cloud

Viele Cloud-Optionen

Wie zuvor beschrieben, profitiert der native SQL-Ansatz beim Thema Cloud davon, dass die SAP-HANA-Plattform sowohl als On-Premise- als auch als Cloud-Variante zur Verfügung steht. Bei der Cloud-Variante SAP HANA Cloud handelt es sich um eine PaaS-Leistung, die die gleichen Möglichkeiten bietet wie die On-Premise-Variante. Strukturell sind beide Varianten gleich aufgebaut, wobei das Gegenstück zum SAP-HANA-XSA-Anwendungsserver die Cloud-Foundry-Technologie ist. Für native Entwicklungen im Rahmen des SAP SQL Data Warehousing ergibt sich hierdurch eine sehr hohe Flexibilität hinsichtlich der Kombination von On-Premise- und Cloud-Lösungen, da diese leicht in die jeweilig andere Umgebung transferiert werden können. Zusätzlich bietet sich die Option der unkomplizierten Integration mit SAP Data Warehouse Cloud.

Self-Service

Universelle Toolauswahl durch SQL

Das SAP SQL Data Warehouse sieht auch bezüglich des Themas Self-Service-BI einige erleichternde Elemente vor. So stehen den Fachanwender*innen durch den einfachen SQL-Zugriff auf die SAP-HANA-Datenbank verschiedene Frontend-Tools zur Datenauswertung und -visualisierung zur Verfügung. Die flexibel gestaltbare Data-Warehouse-Architektur erlaubt zudem das Vorhalten eines Datenbestands, auf den Fachanwender*innen auf der Basis von SQL ohne Hilfe der zentralen IT zugreifen können, um eigene Modelle zu kreieren oder spezifische Data-Science-Methoden anzuwenden. In diesem Szenario kann auch das Erfordernis der Self-Service-Datenbereitstellung gelöst werden. Insgesamt sind durch die systemische Offenheit des SAP SQL Data Warehousing, die es erlaubt, mit individuell präferierten Werkzeugen zu arbeiten, und durch die heute auch unter Fachanwender*innen häufig vorhandenen grundlegenden SQL-Kenntnisse die Hürden für Self-Service-Ansätze vergleichsweise niedrig.

Echtzeit-Analysen

Agile Entwicklung als Prozessbeschleuniger

Im Hinblick auf die zeitgerechte Bereitstellung von Informationen bringt das SAP SQL Data Warehouse die Vorzüge der SAP-HANA-Technologie mit. Die In-Memory-Verarbeitung und die hierdurch möglichen Datenvirtualisierungen machen Echtzeit-Analysen auch im Hinblick auf Big-Data-Szenarien möglich. Darüber hinaus sorgt das Augenmerk auf eine agile Entwicklung des Data Warehouse für Schnelligkeit und zielt auf eine Beschleunigung der Prozesse zur Schaffung von wert- bzw. nutzenstiftenden Ergebnissen bei den Fachanwender*innen. Geschwindigkeit und kurze Durchlaufzeiten sind Kernelemente des SAP SQL Data Warehouse.

Einsatzgebiete

Heterogene Systemlandschaften

Die Kriterien belegen die moderne Ausrichtung des SAP SQL Data Warehousing. Aufgrund der SAP-HANA-Technologie steht es SAP BW/4HANA und SAP Data Warehouse Cloud im Hinblick auf Performance und Leistungsfähigkeit in nichts nach. Zukünftige Big-Data-Szenarien mit IoT- und Social-Media-Anbindung können Sie somit auch mit dieser Form des Data Warehousing realisieren. Im Vergleich zu SAP BW/4HANA ergeben sich durch die Offenheit und Flexibilität des modularen Systems Vorteile in heterogenen Systemlandschaften. Klassische Anwendungsfälle sind Szenarien, in denen DWH-Lösungen von Drittanbietern oder aber auch ein in die Jahre gekommenes SAP BW durch SAP SQL Data Warehouse ersetzt werden sollen. Des Weiteren sind Hybrid-Formen mit einem SAP-BW-System und insbesondere in Verbindung mit SAP Data Warehouse Cloud ein sinnvolles Vorgehen.

Vorteile durch hohe Entwicklungsagilität

Ein zentrales Argument des SAP SQL DWH ist die hohe Entwicklungsagilität, die Sie mit dieser Lösung erreichen können. In Zeiten, in denen der disruptive technologische Fortschritt eine erhöhte Dynamik bezüglich der Adaption neuer Technologien erfordert, sind die kurzen, iterativen und inkrementellen Durchlaufzyklen der DevOps-Philosophie ein Trumpf des SAP SQL Data Warehouse. Die SQL- sowie die Git-Konnektivität erlauben Ihnen die einfache Einbindung verschiedenster DevOps-Tools. Ein Verständnis für vor allem die Bereitschaft zur Anwendung dieser organisatorisch nicht unanspruchsvollen Managementansätze sowie ein solides SQL-Know-how sind Voraussetzung für den erfolgreichen Einsatz von SAP SQL Data Warehousing.

1.3 Warum SAP SQL Data Warehousing?

Hybrid-Formen evaluieren

Der Vergleich der SAP-Data-Warehousing-Ansätze zeigt, dass die Vorgehensweisen und Systeme den Anforderungen an ein modernes Data Warehouse unterschiedlich gerecht werden. Betrachten Sie die Stärken und Schwächen hinsichtlich der aufgeführten Kriterien daher vor dem Hintergrund der konkreten Ausgangssituation und dem beabsichtigten Einsatzgebiet. Von einer grundsätzlichen Überlegenheit einer der drei Ansätze kann nicht gesprochen werden. Vielmehr sind Mischformen und hybride Architekturen denkbar und möglich. Die Leistungsfähigkeit der SAP-HANA-Datenbank bietet hier einen guten Anknüpfungspunkt.

Dennoch gibt es gute Gründe, um sich gegenwärtig und mit Blick auf die Zukunft für das SAP SQL DWH zu entscheiden. Neben den bereits skizzierten Eigenschaften bezüglich der Anforderungen an ein modernes Data Warehouse wollen wir Ihnen abschließend ein spezifisches Bild der Voraussetzungen und der Argumente für SAP SQL Data Warehousing liefern. Damit geben wir Ihnen zudem einen Ausblick auf die weiteren Inhalte des Buches, in denen wir die Vorteile im Detail erläutern.

1.3.1 Vorausetzungen

SAP SQL Data Warehousing ermöglicht Ihnen den schnellen und wirksamen Einsatz der nativen SAP-HANA-Fähigkeiten und ist durch den Fokus auf SQL besonders geeignet, um mit anderen Systemen und Anwendungen zu interagieren. Aus diesem Grund ist es sowohl in Systemlandschaften mit SAP-Einschlag als auch in sonstigen heterogenen Landschaften eine valide Option.

Solides SQL-Know-how von Vorteil

Für alle Szenarien sind ein solides SQL-Know-how und Erfahrungswerte bezüglich der SQL-Entwicklung bei Entwickler*innen ein Vorteil. Ein zwingendes Erfordernis ergibt sich jedoch nicht, da die SAP-HANA-Plattform speziell in SAP Web IDE viele grafische Werkzeuge mitbringt, die eine harte Codierung entbehrlich machen.

Offenheit gegenüber agilen Methoden

Darüber hinaus setzt der Entwicklungsansatz von SAP SQL Data Warehousing zentral auf Methoden der DevOps-Philosophie. Kenntnisse dieser Philosophie und anderer agiler Entwicklungsansätze sind auch hier von Vorteil, aber kein Muss. Lediglich die Bereitschaft zur Anwendung agiler und kollaborativer Managementmethoden ist von Bedeutung, da sich im Vergleich zu bisher gängigen Ansätzen der Data-Warehouse-Entwicklung einige Unterschiede ergeben, die auch von Ihren Teams mitgetragen werden müssen. Bei entsprechendem Engagement kann der Wunsch nach höherer Agilität allerdings sehr gut erfüllt werden, was für Unternehmen, deren Prozesse stark datengetrieben sind und die regelmäßige Anpassungen ihrer Datenstrukturen benötigen, von großem Vorteil ist.

Bereitschaft zur Automatisierung

Im Rahmen des konkreten DevOps-Ansatzes für das SAP SQL Data Warehouse bestehen zudem erhöhte Automatisierungspotenziale. Damit Sie diese nutzen können, ist der Rückgriff auf spezifische Tools notwendig. Sie stehen häufig als Open-Source-Software zur Verfügung und sind in der Welt der Softwareentwicklung seit Jahren etabliert. Dennoch muss auch hier die Bereitschaft vorhanden sein, neue Technologien zu verwenden und insbesondere die übergeordnete Methodik konsequent einzuhalten.

1.3.2 Argumente für SAP SQL Data Warehousing

Das SAP SQL Data Warehousing besticht durch seine Flexibilität im Hinblick auf Architektur, Datenmodellierung und Agilität bezüglich des Entwicklungsansatzes. Durch die Universalität der Datenbanksprache SQL und die hierdurch sichergestellte Offenheit des Systems entsteht ein hoher Freiheitsgrad bei der Gestaltung der Datenstrukturen und Datenmanagement-Prozesse. Tools und Software von Drittanbietern, insbesondere zur Automatisierung der Entwicklungs-Pipeline, können frei gewählt und in die DWH-Landschaft eingebunden werden. Das Komplexitäts-Level der Systemlandschaft ist somit maßgeblich von den Vorstellungen der Anwender*innen bzw. Entwickler*innen abhängig und individuell skalierbar.

SAP SQL DWH erfüllt Bedürfnisse der Anwender*innen

Mit diesen Eigenschaften verfügt SAP SQL Data Warehouse über Stärken, die von BI-Fachleuten hinsichtlich der Modernisierung ihrer Data Warehouses gegenwärtig stark nachgefragt werden. So kommt die aktuelle Forschungsstudie »Modernizing the Data Warehouse: Challenges and Benefits« des Forschungs- und Beratungsinstituts Business Application Center (BARC) zu dem Schluss, dass die wichtigsten Trends im BI-Umfeld auf ein effizienteres, effektiveres und agiles Datenmanagement auf der Basis von Datensteuerung, Automatisierung und Self-Service ausgerichtet sind (vgl. Bloemen, 2019).

Grundlage dieser These ist eine Befragung von knapp 400 überwiegend aus Europa stammenden Fachleuten zu den größten Herausforderungen des Data Warehousing. Zeitraubende Entwicklungsprozesse und die nur eingeschränkte Unterstützung von Self-Service-BI werden dort als größte Hürden genannt (siehe Abbildung 1.5).

Entwicklungsprozesse beschleunigen

Darüber hinaus wird die Unfähigkeit zur Anpassung an neue Anforderungen/Änderungen als maßgebliches Hindernis auf dem Weg zu einem effizienteren Datenmanagement betrachtet. Insgesamt tauchen die Wörter *agil* und *Agilität* in der 25-seitigen Studie 30-mal auf, was den Wunsch der Anwender*innen in dieser Hinsicht nachdrücklich betont. Genau diese geforderte Entwicklungsagilität ist das Kernargument des SAP SQL Data Warehousing. Der gesamte Prozess ist auf die DevOps-Philosophie ausgerichtet und verfolgt das Ziel, insbesondere die Entwicklung des DWH durch paralleles Arbeiten und kurze iterative und inkrementelle Durchlaufzyklen deutlich zu beschleunigen.

Abbildung 1.5 Gegenwärtige Herausforderungen im Data-Warehouse-Umfeld (Quelle: Bloemen, 2019)

SAP BW/HANA in Kombination mit SAP SQL DWH

Ein weiterer Beleg für die Nutzung der SAP-HANA-Plattform zum SAP SQL Data Warehousing ist eine von SAP gesponsorte Untersuchung der Beratungshäuser Eckerson Group und BARC zur Kundenzufriedenheit mit SAP BW/4HANA aus dem Jahr 2019 (vgl. Wood, Grosser, 2019). Darin bescheinigen die vier befragten Unternehmen SAP BW/4HANA, eine zufriedenstellende und moderne Data-Warehouse-Lösung zu sein. Eine überwiegende Schlussfolgerung der Unternehmen, die letztlich auch von den Analyst*innen geteilt wird, ist es jedoch, sich künftig mehr auf die nativen Möglichkeiten der SAP-HANA-Plattform und damit auf SAP SQL Data Warehousing zu fokussieren und so das von ihnen gewählte SAP-BW/4HANA-System noch umfassender zu entwickeln. Die beschriebene Offenheit für verschiedene Visualisierungstools und die Flexibilität der Gestaltung sind dabei die ausschlaggebende Motivation.

DevOps als Katalysator

Unter Berücksichtigung der Voraussetzungen und vor dem Hintergrund der gegenwärtigen und absehbaren Anforderungen spricht vieles dafür,

sich für SAP SQL Data Warehousing zu entscheiden. Das zentrale Kernkonzept DevOps und die weiteren Vorzüge, die sich um diesen Gedanken ergeben, lassen sich in die folgenden Einzelaspekte gliedern:

- **Grafische Editoren**
  Bei der Entwicklung der Data-Warehouse-Architektur können Sie auf die grafischen Editoren der webbasierten Entwicklungsumgebung SAP Web IDE sowie auf SAP PowerDesigner zurückgreifen. Der SQL-Code wird bei diesen Werkzeugen effizient im Hintergrund erzeugt. Diese Vorgehensweise verringert die Komplexität und erleichtert den Einstieg in die Data-Warehouse-Entwicklung.

Grafische Editoren

Auf die Werkzeuge SAP Web IDE und SAP PowerDesigner gehen wir im Überblick in Abschnitt 2.3, »Werkzeuge der SAP-HANA-Plattform«, ein. Einzelheiten und praktische Anleitungen zu den grafischen Editoren erwarten Sie bezüglich der Modellierung der Datenmodelle in Kapitel 7, »Modellierung des konzeptionellen Datenmodells«, und in Kapitel 8, »Modellierung der physischen Datenmodelle«. In Kapitel 9, »Entwicklung des SQL Data Warehouse«, erfahren Sie, wie Sie ETL-Strecken und virtuelle Objekte grafisch erstellen.

- **Modellierungsfreiheit**
  Die Datenmodellierung kann nach verschiedenen Formen wie 3NF dimensional oder per Data Vault erfolgen. Diese Freiheit erfährt eine immer größere Nachfrage. Speziell die Data-Vault-Modellierung, die auch wir Ihnen für die Modellierung des SAP SQL Data Warehouse empfehlen, hat in den letzten Jahren an Bedeutung gewonnen.

Modellierung

Allgemeine Informationen zum Thema Modellierung eines Data Warehouse haben wir für Sie in Kapitel 5, »Methodische Grundlagen für das Data Warehousing«, zusammengetragen. Hier erfahren Sie alles über die verschiedenen Modellformen und ihre Vorzüge. In Kapitel 7, »Modellierung des konzeptionellen Datenmodells«, und in Kapitel 8, »Modellierung der physischen Datenmodelle«, erläutern wir die praktische Durchführung der Modellierung mit den entsprechenden Werkzeugen.

- **Versionierung**
  Der Quellcode für jegliche Applikations- und Datenbankartefakte wird in Git gespeichert, was die Möglichkeit einer parallelen Entwicklung stark

verbessert. Git hat sich in der Softwareentwicklung als führende Versionsverwaltung etabliert und birgt auch für die Entwicklung des SAP SQL Data Warehouse Vorteile bezüglich der Beschleunigung der Entwicklungsprozesse.

Versionierung

Grundlegende Informationen zum Thema Quellcode-Verwaltung mit Git finden Sie in Abschnitt 5.4.1, »Quellcode-Verwaltung mit Git«, und in Abschnitt 9.1, »Initialisierung von Git und SAP Web IDE«.

- **Continuous Integration und Continuous Testing**
  Continuous Integration (CI) und Continuous Testing (CT) sind zwei der vier essenziellen Prozesse der DevOps-Philosophie, die ein hohes Maß an Agilität und Qualität ermöglichen.
- **Continuous Delivery und Continuous Deployment**
  Continuous Delivery und Continuous Deployment sind die weiteren zentralen Prozesse der DevOps-Philosophie, durch die die Auslieferungszyklen von nutzenstiftenden Entwicklungen beschleunigt werden, um die Time-to-Value bzw. Time-to-Market erheblich zu verkürzen.
- **Flexibles und verteiltes Deployment**
  Entwicklungsartefakte können flexibel und auf der Basis von Software-Containern verteilt ausgeliefert werden. Dies ist sowohl in der Cloud als auch on-premise möglich. Durch die Container-Technologie ist es zudem möglich, verschiedene Applikationen auf einer Datenbank zu platzieren.
- **Automation**
  Durch die DevOps-Prozesse können Sie die Auslieferung von Anwendungs- und Datenbankartefakten weitgehend automatisieren, um die Durchlaufgeschwindigkeit und die Qualität weiter zu steigern.

DevOps

Abschnitt 4.1.3, »DevOps«, führt Sie in das Thema DevOps ein und erläutert die grundlegenden Ziele und Charakteristika sowie die Herausforderungen und Vorteile dieser Philosophie. In Abschnitt 4.2.1, »Klassische DWH-Entwicklung vs. DevOps«, finden Sie Details zu den Kontinuitätsprozessen im konkreten Zusammenhang mit der Entwicklung von SAP SQL Data Warehouse. Das Thema Automation von Entwicklungsprozessen erläutern wir detailliert in Abschnitt 5.4.3, »Automatisierung von Entwicklungsprozessen«.

- **Datengetriebene Applikationen**
  Insgesamt wird auf diese Weise die Erstellung von datengetriebenen, geschäftsspezifischen Multimodell-Anwendungen erleichtert.

Im weiteren Verlauf des Buches werden wir diese Aspekte weiter vertiefen und praxisnah erläutern.

1.4 Zusammenfassung

In diesem Kapitel haben wir die grundlegenden Anforderungen an das moderne Data Warehousing vorgestellt. Auf Basis dieser Anforderungen haben wir die drei Data-Warehousing-Ansätze SAP BW/4HANA, SAP Data Warehouse Cloud und SAP SQL Data Warehousing gegenübergestellt und einen Vergleich vorgenommen. Alle Produkte beinhalten Werkzeuge für die verschiedenen Kriterien; allerdings haben sie unterschiedliche Stärken. Für eine strategische Ausrichtung können Hybrid-Konstellationen eine vollumfängliche Lösung darstellen. Sie kennen nun die Argumente für das SAP SQL Data Warehousing und die native Nutzung der SAP-HANA-Datenbank. Dieser Ansatz verfügt aufgrund seines Fokus auf Agilität und die vorhandene strukturelle Flexibilität und systemische Offenheit über Eigenschaften, die Ihnen helfen, künftigen analytischen Problemstellungen im Unternehmenskontext zu begegnen.

Kapitel 10

Deployment des SAP SQL Data Warehouse

»Focus on being productive instead of busy.«
Tim Ferriss

In diesem Kapitel verlassen wir im DevOps-Prozessmodell den Bereich der Designtime und fokussieren uns mit den Schritten **Release** und **Deploy** auf die produktive Auslieferung des SAP SQL DWH (siehe Abbildung 10.1). Für unseren Beispielfall haben wir einen umfangreichen Stand über alle Ebenen des DWH in SAP Web IDE entwickelt und wollen diesen nun exemplarisch nutzen.

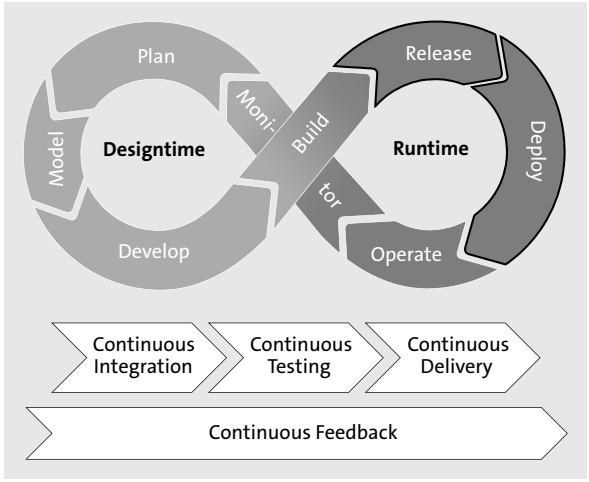

Abbildung 10.1 Auslieferungsphase im DevOps-Prozessmodell

In der Praxis lassen sich bereits kleinere Arbeitsstände im Sinne von DevOps und der agilen Entwicklung kontinuierlich in das operative Data Warehouse integrieren. Der Entwicklungs- und Auslieferungsrhythmus kann hier individuell bestimmt werden. Die Flexibilität ergibt sich durch den in Kapitel 9, »Entwicklung des SQL Data Warehouse«, praktisch geschilderten modularen Aufbau des SAP SQL DWH in Containern der SAP HANA Deployment Infrastructure, die prinzipiell darauf ausgerichtet ist, Multi-Target Ap-

plications (MTA) zu entwickeln und auf dem SAP-HANA-XSA-Anwendungsserver oder der Cloud-Anwendungsplattform Cloud Foundry zu betreiben (siehe Abschnitt 2.2.4, »Anwendungsentwicklung«, und Abschnitt 6.1, »Infrastruktur«). Nachfolgend zeigen wir Ihnen, welche Vorteile diese Struktur für die Auslieferung des SAP SQL DWH, insbesondere für die Automatisierung der Prozesse, bietet. Wir starten dazu in Abschnitt 10.1 mit den manuellen Möglichkeiten eines Deployments, durch die grundlegenden Mechanismen deutlich werden. In Abschnitt 10.2 erläutern wir im Anschluss die automatische Vorgehensweise auf der Basis von Git und Jenkins. Zuletzt gehen wir in Abschnitt 10.3 auf einzelne Aspekte der Testautomation in diesem automatischen Szenario ein.

10.1 Manuelles Deployment

Zwei Optionen für das manuelle Deployment

Beim manuellen Deployment fassen Sie in SAP Web IDE in Verbindung mit dem Git Repository die Multi-Target Application in einem MTA-Archiv mit der Endung **.mtar** zusammen, das anschließend auf dem SAP-HANA-XSA-Anwendungsserver oder auf einer Cloud-Foundry-Plattform ausgeführt wird und die Datenbankobjekte in SAP HANA umsetzt (siehe Abbildung 10.2). Die einzelnen Schritte erläutern wir Ihnen nachfolgend praktisch anhand unseres Beispielfalls.

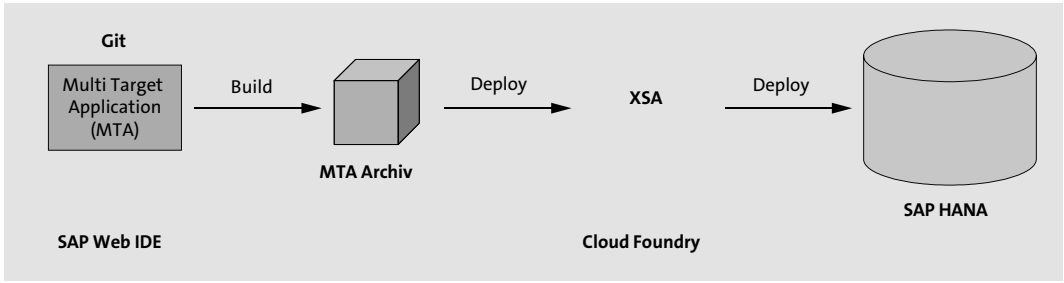

Abbildung 10.2 Manueller Deployment-Prozess

10.1.1 Erstellen des MTA-Archivs

Transporteinheit MTA-Archiv

Ein MTA-Archiv (**.mtar**) ist eine Zusammenstellung (Package) der verschiedenen Module der Multi-Target Application, die in unserem Fall des SAP SQL DWH grundsätzlich nur aus Datenbankmodulen besteht. Neben diesen Modulen enthält das Archiv noch den Ordner **META-INF**. Dieser beinhaltet zum einen die Datei **MANIFEST.MF**, die die Organisation der Module im Archiv definiert, sowie zum andern die Datei **mtad.yaml**, die das Zusammenspiel der Module und Services, das in der Datei **mta.yaml** beschrieben ist, um spezifische Deployment-Informationen ergänzt.

Versionierung in mta.yaml

Die Versionierung des MTA-Archivs wird über den Parameter `version:` in der Datei **mta.yaml** gesteuert.

Das MTA-Archiv erzeugen Sie in SAP Web IDE über den Basis-Projektordner **Demo** und die Befehlsfolge **Build • Build** (siehe Abbildung 10.3). Je nach Größe der Applikation kann der Build-Vorgang einige Minuten in Anspruch nehmen.

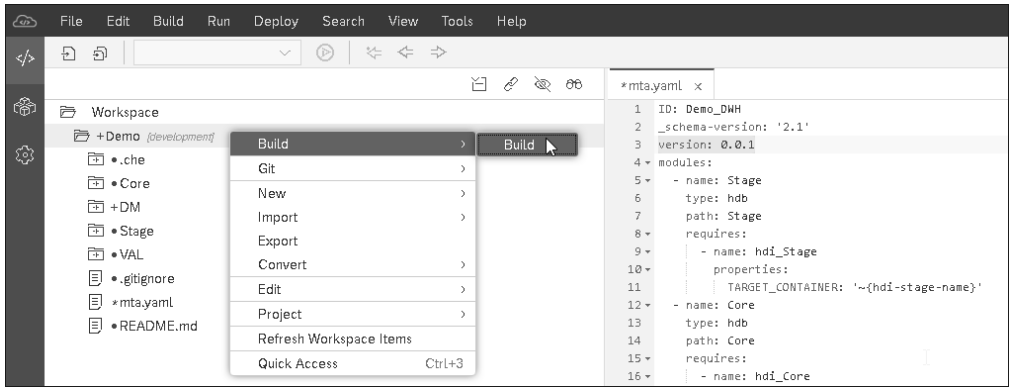

Abbildung 10.3 Build des MTA-Archivs

Im Anschluss findet sich im Projekt ein weiterer Ordner **mta_archives**, der je nach Nutzung der Versionierung in der Datei **mta.yaml** mehrere **.mtar**-Dateien enthält (siehe Abbildung 10.4).

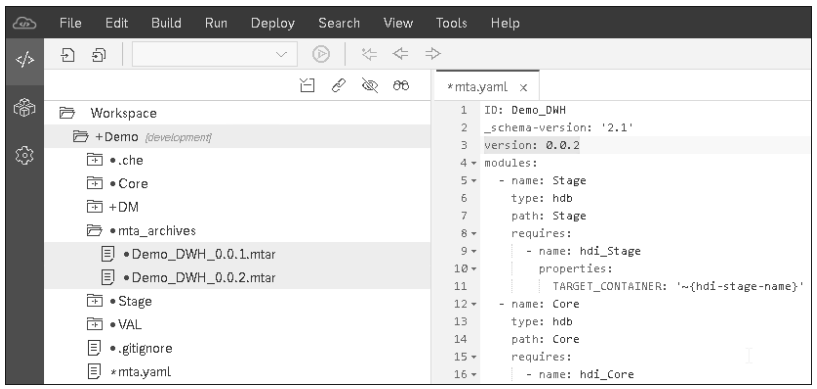

Abbildung 10.4 Versionierung in der Datei mta.yaml

10.1.2 Deployment des MTA-Archivs mit SAP Web IDE

Manuelles Deployment mit SAP Web IDE

Liegt das MTA-Archiv vor, können Sie es in SAP Web IDE über den Befehl **Deploy** und die beiden Optionen **Deploy to SAP Cloud Platform** und **Deploy to XS Advanced** in einen Space nach Wahl übertragen (siehe Abbildung 10.5). Durch die entsprechenden Spaces können Sie die Transporte in Entwicklungs-, Test- und Produktivumgebungen gestalten und eine Auslieferungskette erstellen.

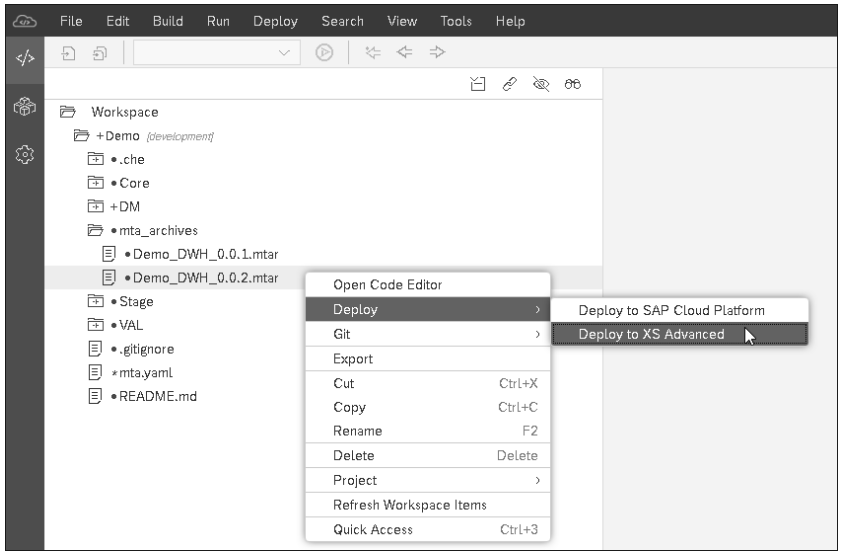

Abbildung 10.5 Deployment-Befehl zu einem MTA-Archiv

10.1.3 Deployment des MTA-Archivs mit dem XSA Client

Manuelles Deployment mit XSA Client

Eine weitere Möglichkeit zum Deployment des MTA-Archivs auf dem SAP-HANA-XSA-Anwendungsserver bietet der *XSA Client*, mit dem der Prozess in einem Command Line Interface durchlaufen werden kann. Der Client kann im *SAP One Support Launchpad* unter *http://s-prs.de/v781733* für verschiedene Betriebssysteme heruntergeladen und über die Konsole geöffnet werden. Die **.mtar**-Datei müssen Sie für dieses Vorgehen aus SAP Web IDE exportieren und lokal speichern. Anstelle des MTA-Archivs können Sie aber auch eine Ordnerstruktur, insbesondere die Git-Struktur, verwenden und über den XSA Client auf dem SAP-HANA-XSA-Anwendungsserver ausführen. Abbildung 10.6 zeigt die Argumente und Optionen des wesentlichen Befehls `xs deploy`. Die Argumente zur Verwendung eines Git Repositorys als Quelle sind dabei markiert. Im Vergleich zu SAP Web IDE bietet der Client bereits mehr Optionen zur Gestaltung des manuellen Deployment-Prozesses und ebnet vom Verfahren den Weg zu automatisierten Prozessen, speziell auf der Basis von Git.

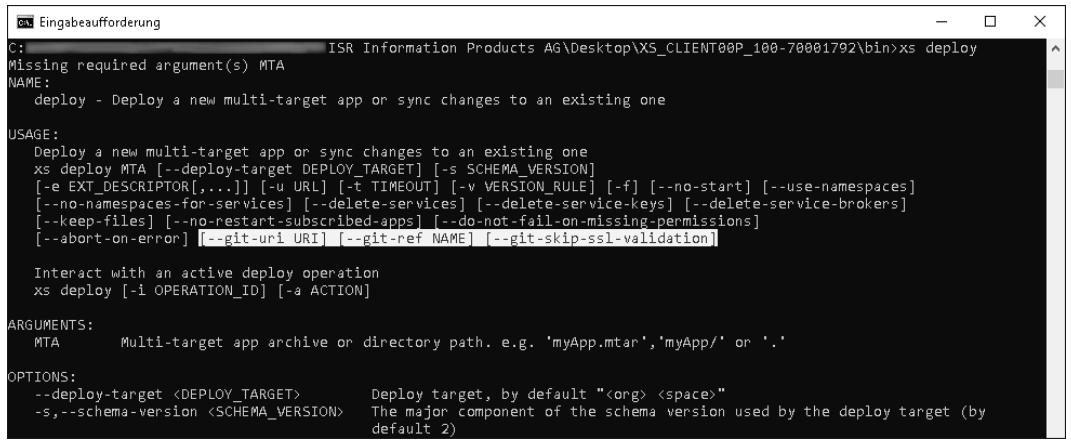

Abbildung 10.6 Optionen und Argumente des XSA Clients

10.1.4 Manuelle Auslieferungskette

Auslieferungskette in SAP HANA XSA und Cloud Foundry

Wie bereits angeklungen, erfolgen beim manuellen Deployment auch die in unserem DevOps-Prozessmodell auf der Runtime-Seite beschriebenen Auslieferungsprozesse manuell. Abbildung 10.7 zeigt das Vorgehen noch einmal plastisch. Das MTA-Archiv wird hier aus einer Entwicklungsumgebung exportiert und in eine Test- bzw. Release- und eine Produktivumgebung importiert. Bei diesen Umgebungen handelt es sich um SAP HANA XSA bzw. Cloud Foundry Spaces, die die Objekte des SAP SQL DWH auf der SAP-HANA-Datenbank isolieren.

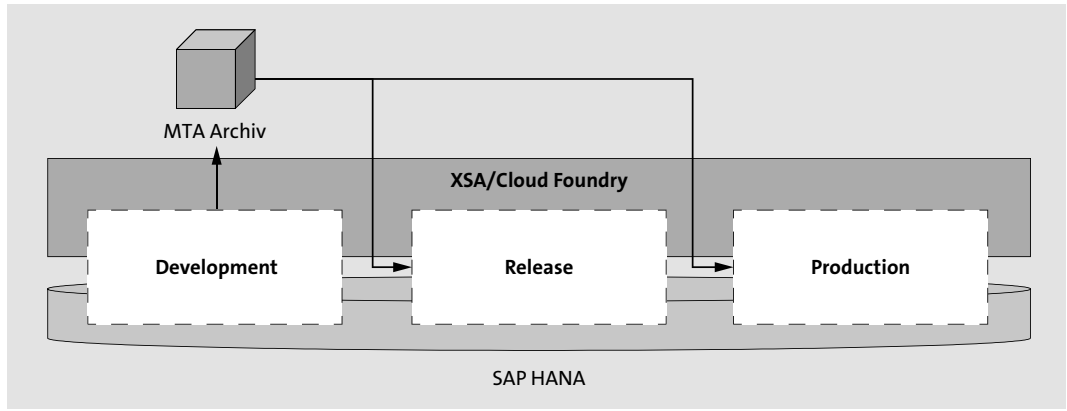

Abbildung 10.7 Auslieferung des MTA-Archivs in verschiedene Spaces

10.2 Automatisches Deployment

Continuous Delivery Pipeline

Zu Beginn dieses Kapitels haben wir, wie in den übrigen Kapiteln zur praktischen Umsetzung des SAP SQL DWH, unser DevOps-Prozessmodell vorangestellt, das wir in Kapitel 4, »Entwicklungsansatz für das SAP SQL DWH«, erarbeitet haben. In Kapitel 4 haben wir das automatische Deployment mithilfe von Git und Build-Server-Produkten als Regelfall beschrieben und die Vorteile dieses Vorgehens umfassend erläutert.

Automatisches Deployment mit Git und Jenkins

In diesem Abschnitt wollen wir die praktische Seite der Methodik vertiefen und Ihnen auf Basis unseres Beispielfalls einen Einblick in den Umgang mit Git und dem bekannten Open-Source-Build-Server Jenkins geben. Das manuelle Deployment war dazu eine gute Einführung, da die Vorgehensweise über den XSA Client letztlich in einer Jenkins-Pipeline automatisiert wird. Abbildung 10.8 zeigt den grundsätzlichen Aufbau einer solchen Deployment-Pipeline. Jenkins ist hier so konfiguriert, dass das Programm spezifische Branches wie Release und Master auf Änderungen hin überprüft. Liegen solche Änderungen durch einen Commit vor, entnimmt Jenkins einen Clone des Source-Codes und verpackt die Bestandteile durch einen Build-Vorgang in einem MTA-Archiv. Dieses wird anschließend über SAP HANA XSA oder Cloud Foundry in der SAP-HANA-Datenbank umgesetzt.

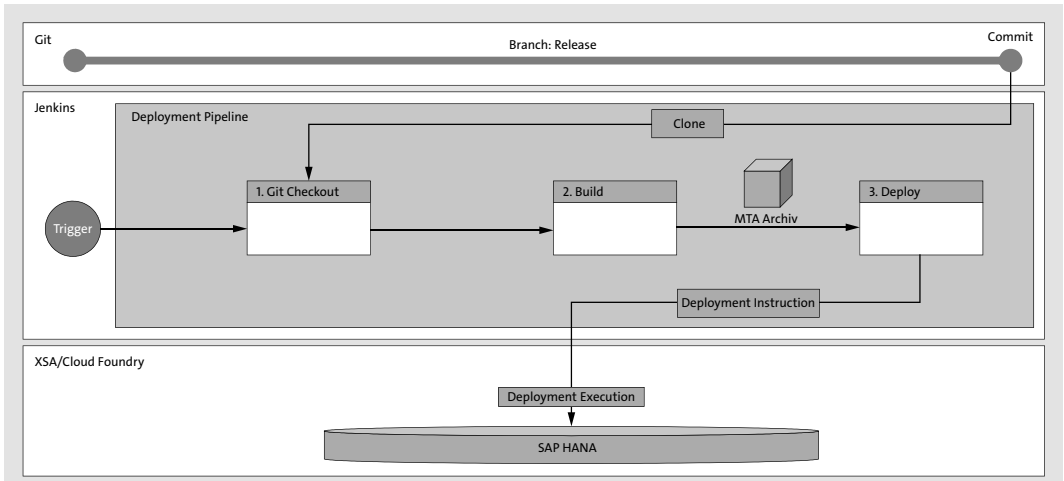

Abbildung 10.8 Automatisierter Deployment-Prozess

Konfiguration in einer Jenkinsfile

Der wesentliche Teil der Konfiguration findet entsprechend der Prozessabbildung in Jenkins statt. In unserem Beispielfall nutzen wir dazu eine **.txt**-Datei mit dem Namen **Jenkinsfile**, mit der wir die Pipeline definieren und

die wir in Git speichern. Die Definition kann jedoch auch über die grafische Oberfläche von Jenkins vorgenommen werden, wodurch jedoch die Vorteile der Versionskontrolle in Git entfallen. Abbildung 10.9 zeigt den Inhalt der Jenkinsfile mit einem simplen Aufbau aus Basisvariablen der SAP-HANA-XSA-Umgebung, einen Build-Block, in dem der Code aus dem definierten Git Branch ausgecheckt und in einem MTA-Archiv zusammengefasst wird, und einen Deploy-Block, in dem das MTA-Archiv durch die XSA-Client-Befehle im definierten Space umgesetzt wird.

master · Demo / Jenkinsfile · Go to file

ISR-SAP-HANA-SQL-DWH Add files via upload · Latest commit b04097e 4 minutes ago · History

1 contributor

75 lines (69 sloc) | 2.99 KB · Raw · Blame

```
pipeline {
    agent any

    environment {
        HANA_XSA_CREDS      = credentials('XSA_JENKINS')
        XSA_API_ENDPOINT    = 'https://awshn2.isr.local:30030/'
        ORGANIZATION        = 'ISR'
        CI_SPACE            = 'DEMO'
        modulePaths         = sh(script: '''awk -F: '$1 ~ /path/ { gsub(/\\s/,"", $2); print $2 }' ${WORKSPACE}/mta.yaml''', returnStdout: true)
        mtaName             = sh(script: '''awk -F: '$1 ~ /^ID/ { gsub(/\\s/,"", $2); print $2 }' ${WORKSPACE}/mta.yaml''', returnStdout: true).trim()

    }
    stages {
        stage('Build') {
                    when {
                expression {env.BRANCH_NAME == 'release'}
            }
            steps {
                echo 'Building..'
                // create local npmrc file
                    sh('''cat <<EOF > .npmrc
                    registry=https://registry.npmjs.org/
                    @sap:registry=https://registry.npmjs.org
                    EOF''')

                // add .npmrc to the modulePaths
                    sh('''for path in ${WORKSPACE}/${modulePaths}; do
                        if [ -d "$path" ];
                            then ln -sft $path ../.npmrc
                        fi
                        done''')

                // build mtar package
                    sh('/data/SAP/mbt/mbt build -p=xsa --mtar ${mtaName}.mtar -m=verbose')
            }
        }
        stage('Deploy') {
                    when {
                expression {env.BRANCH_NAME == 'release'}
            }
            steps {
                        echo 'Deploying....'
                        sh('/data/SAP/xs_client/bin/xs api $XSA_API_ENDPOINT --cacert /data/SAP/xs_client/xsa_api.cer')
                        sh('/data/SAP/xs_client/bin/xs login -u $HANA_XSA_CREDS_USR -p $HANA_XSA_CREDS_PSW -o $ORGANIZATION -s $CI_SPACE')
                        sh('/data/SAP/xs_client/bin/xs deploy -f ${WORKSPACE}/mta_archives/${mtaName}.mtar')
```

Abbildung 10.9 Pipeline in einer Jenkinsfile konfigurieren

In dieser einfachen Form ist als Quelle im Git lediglich der Release-Branch hinterlegt. Jenkins hört diesen kontinuierlich ab und startet im Falle von Veränderungen die definierte Deployment-Pipeline. Eine Erweiterung um entsprechende Szenarien für Development und Production Spaces mit den entsprechenden Git Branches Development und Master ist allerdings leicht möglich.

Pipeline-Übersicht in Jenkins Blue Ocean

Über die grafische Benutzeroberfläche können Sie nach der Definition im Bereich **Blue Ocean** die Ausführungs-Pipeline in übersichtlicher Form überprüfen. Abbildung 10.10 zeigt die Übersicht der Pipelines pro Branch, die in unserem Fall nur mit der Release-Pipeline gefüllt ist. Öffnen Sie diese, sehen Sie die Ergebnisse des letzten Pipeline-Durchlaufs. Es wurden die von uns in der Jenkinsfile definierten Schritte **Build** und **Deploy** mit ihren etwaigen Befehlen (siehe Abbildung 10.11 und Abbildung 10.12) ausgeführt, und unser kleines Beispiel-SAP-SQL-DWH konnte in ca. fünf Minuten auf dem SAP-HANA-XSA-Anwendungsserver und in der SAP-HANA-Datenbank umgesetzt werden.

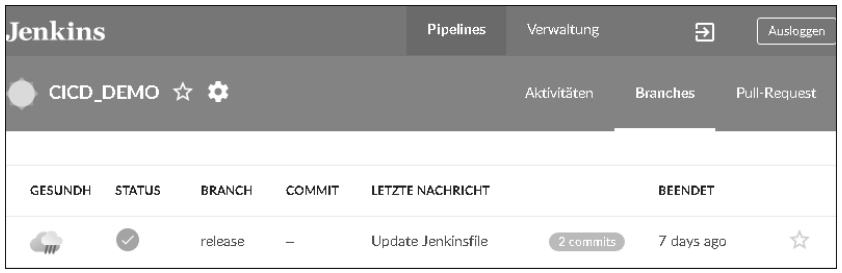

Abbildung 10.10 Pipeline-Übersicht in Jenkins Blue Ocean

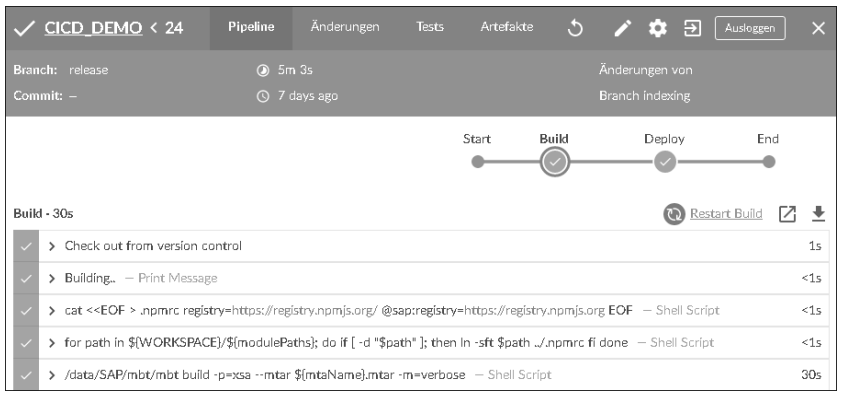

Abbildung 10.11 Prozessschritt »Build«

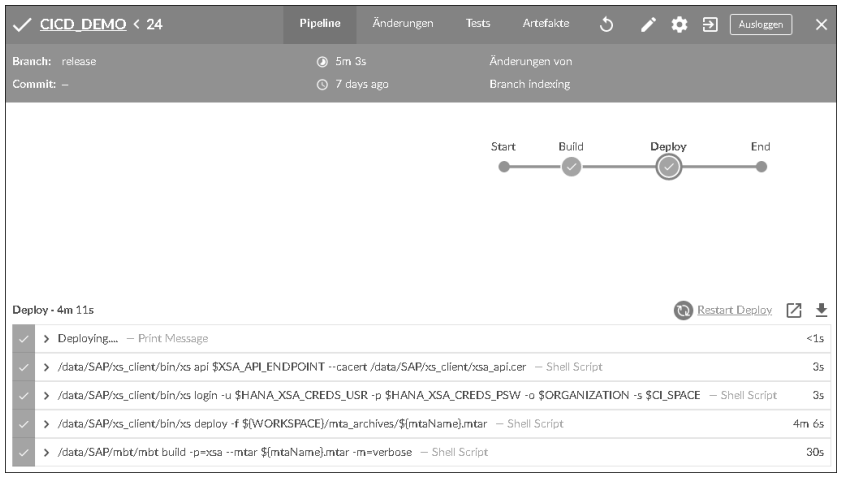

Abbildung 10.12 Prozessschritt »Deploy« mit XSA-Client-Befehlen

Verzahnung mit Jira für die zentrale Projektkontrolle

Durch entsprechende organisatorische Rahmenbedingungen wie die Verwaltung der Git Branches mit Freigabeprozessen bei Merge Requests und die Integration von Issue-Tracking-Lösungen wie Jira im Git Repository (siehe Abbildung 10.13), mit denen Sie Nachrichten im Falle von Commits und Merge Requests zentral zusammenfassen können, wird das Szenario des Continuous Deliverys oder sogar des Continuous Deployments für das SAP SQL DWH greifbar.

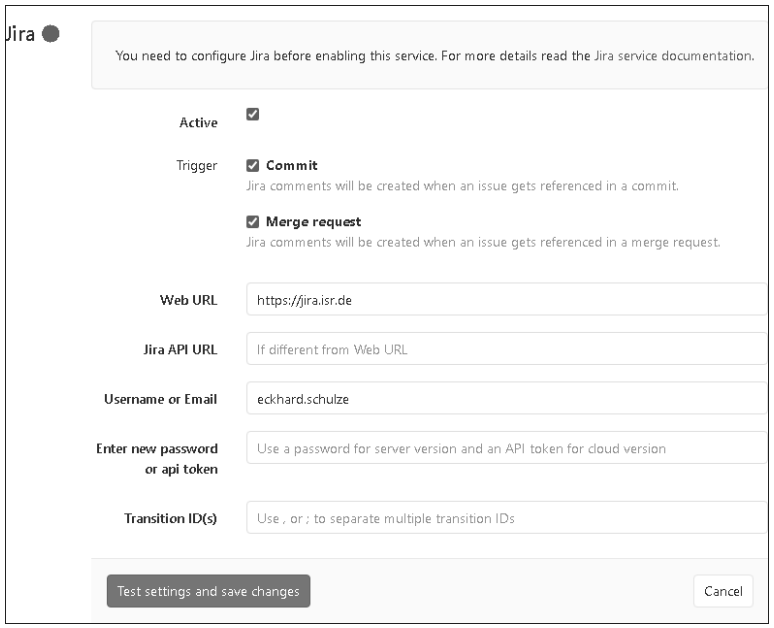

Abbildung 10.13 Integration von Jira im Git Repository

Bei all den manuellen Überprüfungsmöglichkeiten, die das Git Repository durch Codevergleiche bereits ermöglicht, ist für die Continuous-Prozesse vor allem auch die Automatisierung von Tests entscheidend. Auf diesen wichtigen Aspekt des Deployments gehen wir daher in nächsten Abschnitt ein.

Jenkinsfile mit kommentiertem Code

Die beispielhafte Jenkinsfile finden Sie ebenfalls auf der Webseite zum Buch unter *www.sap-press.de/5167* sowie im Git Repository zum Buch unter *https://github.com/ISR-SAP-HANA-SQL-DWH/Demo*. Hier haben wir in der Readme-Datei weitere Informationen hinterlegt, mit der Sie eine Jenkins-Deployment-Pipeline auch für Ihre Systemumgebung aufbauen können. Grundsätzliche Informationen zur Konfiguration können Sie zudem in der Jenkins-Dokumentation unter *http://s-prs.de/v781731* nachschlagen.

10.3 Testautomation

Integration von Continuous Testing

Der Mehrwert des Continuous-Testing-Prozesses haben wir bereits in Abschnitt 4.2.3, »Kontinuierliche Prozesse«, dargelegt. Abbildung 10.14 zeigt eine Möglichkeit, wie automatisierte Tests, die eine wichtige Rolle in diesem Zusammenhang spielen, in die im letzten Abschnitt vorgestellte automatisierte Deployment-Pipeline des SAP SQL DWH integriert werden können. So lassen sich insbesondere Tests in Bezug auf Metadaten bereits auf der Basis des Clone, den Jenkins vom definierten Branch erstellt, zwischen den Entwicklungen in SAP Web IDE und den Datenmodellen in SAP PowerDesigner vergleichen. Gibt es beispielsweise unterschiedlich viele Entitäten und Attribute, weichen Bezeichnungen oder Datentypen voneinander ab oder sind weitere Aspekte nicht identisch, wird der Deployment-Vorgang abgebrochen, und die Abweichungen müssen zunächst behoben werden.

Tests zu technischen Aspekten, wie beispielsweise die Funktionsfähigkeit von Calculation Views und Flowgraphs, lassen sich besser nach dem Deployment prüfen. Dies gilt auch für Tests zu Dateninhalten, die mit SQL-Abfragen gegen die Inhalte der SAP-HANA-Datenbank überprüft werden. In diesem Zusammenhang können auch Aspekte des Datenschutzes und der Datensicherheit getestet werden, wie z. B. die korrekte Maskierung von Daten und die Zugriffsmöglichkeiten von Datenbank-Usern.

Neben dieser grundsätzlichen Aufteilung in Tests vor und nach dem Deployment kann durch das Vorhalten verschiedener Tests für Entwicklungs-, Test- und Produktivumgebungen die Effektivität der Auslieferungskette gesteigert werden. Es gilt das Iterationsprinzip mit einem Fokus auf frühzeitige Fehlererkennung, durch das in kleinen sukzessiven Schritten schneller werthaltige Ergebnisse erreicht werden können. Für unseren Beispielfall wollen wir Ihnen einen kleinen nachgelagerten Test demonstrieren.

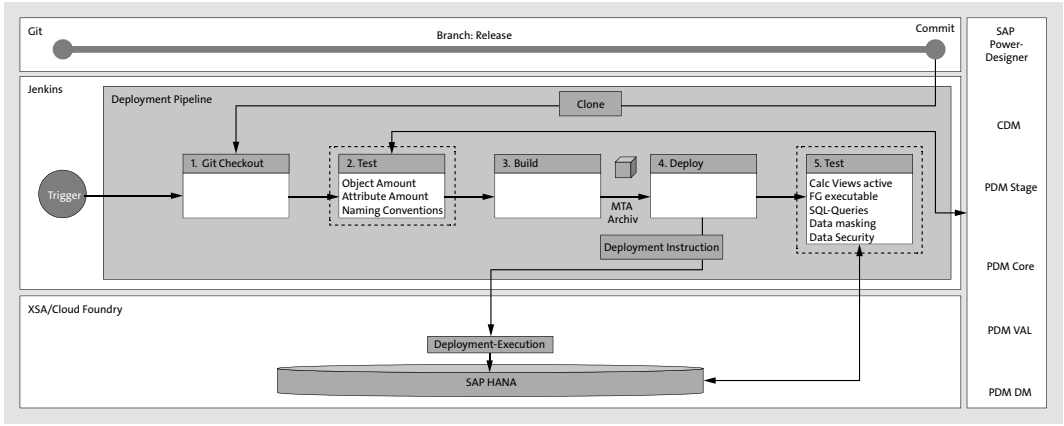

Abbildung 10.14 Automatisiertes Deployment mit Testautomation

Definition in der Jenkinsfile

Die Integration von Tests in die Deployment-Pipeline erfolgt hierbei ebenfalls in der Jenkinsfile. Hier können Sie beliebig viele Tests als Prozesselement (`stage`) einbinden. Abbildung 10.15 zeigt unseren Test für den Release-Branch nach dem Deployment, für den am Anfang der Jenkinsfile weitere Umgebungsvariablen definiert sind.

In unserem Fall handelt es sich um ein Testpaket auf der Basis der JavaScript-Lösung *npm*, das wir in einem separaten Git Repository deponiert haben. In diesem nutzen wir die Testbibliotheken *Mocha* und *Chai*, um Metadaten der Calculation Views des Datenbankmoduls Virtual Analytical Layer auf ihre Richtigkeit hin zu prüfen.

Testpaket mit erklärtem Code zur freien Verfügung

Das hier verwendete Testpaket zur einfachen Überprüfung der Namenkonventionen der DWH-Schicht Virtual Analytical Layer finden Sie im Git Repository unter *https://github.com/ISR-SAP-HANA-SQL-DWH/Demo_TA*. Auch hier haben wir in der Readme-Datei einige weitere Informationen, die es Ihnen ermöglichen sollten, das Testszenario auch für Ihre Systemlandschaft aufzubauen und automatische Tests im Deployment-Vorgang des SAP SQL DWH zu integrieren.

```
        stage('Deploy') {
                        when {
                expression {env.BRANCH_NAME == 'release'}
            }
            steps {
                        echo 'Deploying....'
                        sh('/data/SAP/xs_client/bin/xs api $XSA_API_ENDPOINT --cacert /data/SAP/xs_client/xsa_api.cer')
                        sh('/data/SAP/xs_client/bin/xs login -u $HANA_XSA_CREDS_USR -p $HANA_XSA_CREDS_PSW -o $ORGANIZATION -s $CI_SPACE')
                        sh('/data/SAP/xs_client/bin/xs deploy -f ${WORKSPACE}/mta_archives/${mtaName}.mtar')
                }
        }
        stage('Test') {
            steps {
                git credentialsId: 'ff9118fa-4ef6-4fb2-a357-9b40114b6683',
                    url: 'https://github.com/ISR-SAP-HANA-SQL-DWH/Demo_TA.git'

                dir("/var/lib/jenkins/workspace/CICD_DEMO_development/Demo_TA"){
                    sh('''
                        export NVM_DIR="$HOME/.nvm"
                        [ -s "$NVM_DIR/nvm.sh" ] && \\. "$NVM_DIR/nvm.sh"
                        [ -s "$NVM_DIR/bash_completion" ] && \\. "$NVM_DIR/bash_completion"
                        nvm use --lts
                        npm install
                        npm test
                    ''')
                }
            }
        }
    }
}
```

Abbildung 10.15 Testelement in Jenkinsfile

Anstelle von JavaScript-Testautomation-Frameworks können Sie hier aber auch entsprechende Python- oder sonstige Skriptlösungen nutzen, solange sie gut mit den grundlegenden Voraussetzungen einer Jenkins-Deployment-Pipeline harmonieren. Die Testergebnisse werden im Anschluss wiederum in Jenkins Blue Ocean übersichtlich dargestellt. Abbildung 10.16 zeigt das im Ablauf ergänzte Testelement der Jenkinsfile mit den einzelnen Informationen zur Ausführung des Testpakets.

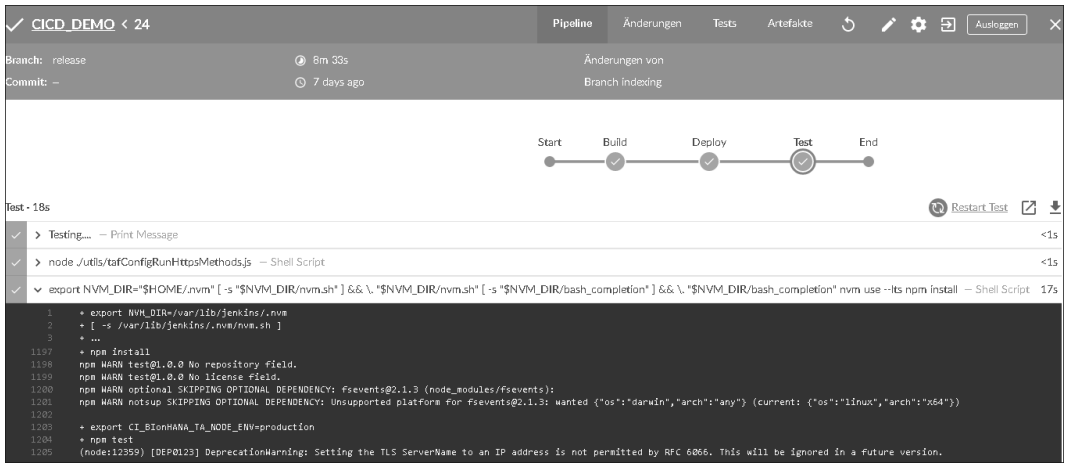

Abbildung 10.16 Testmodul in Blue Ocean

Nutzung des SAP Cloud Application Programming Model

Dieses einfache Prinzip der Nutzung von Testskripten in Paketlösungen, die über die Jenkins-Deployment-Pipeline in die Auslieferungskette integriert werden, lässt sich beliebig weiter ausformen.

So bietet beispielsweise das SAP Cloud Application Programming Model eine Möglichkeit, um das Verfahren in eine ansprechendere visuelle Form zu bringen und die Verwaltung der Testautomation für das SAP SQL DWH zu erleichtern.

Abbildung 10.17 zeigt hier eine mögliche Aufbereitung zur Darstellung der Ergebnisse von zwei Testmodulen, die Vergleiche zwischen den Datenmodellen in SAP PowerDesigner und den Data-Warehouse-Objekten in SAP HANA vornehmen.

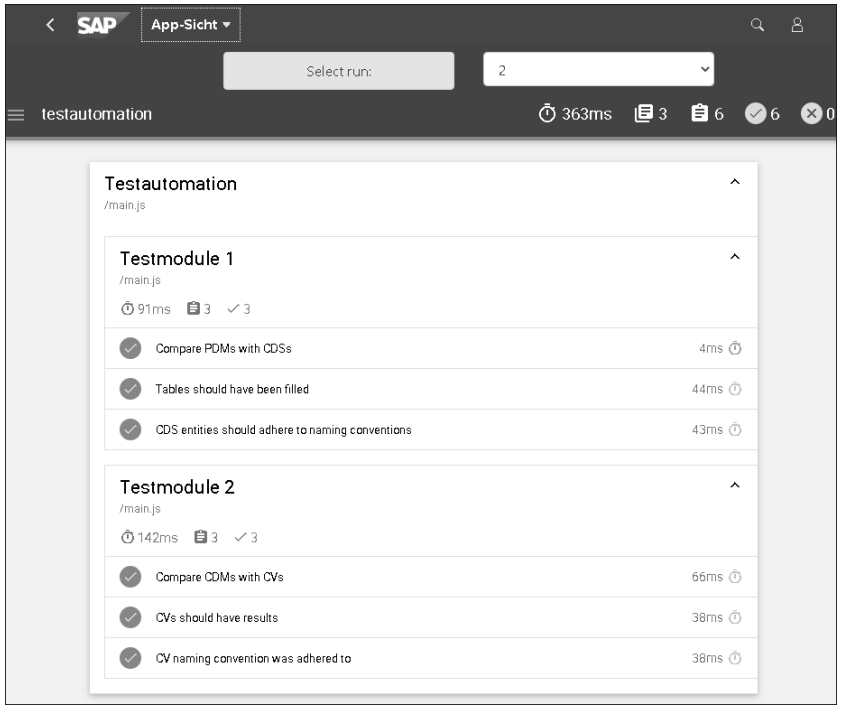

Abbildung 10.17 Mögliche Aufbereitung im SAP Cloud Application Programming Model

Die Prozesse des Continuous Testing und Continuous Delivery können durch solche Ansätze weiter vereinfacht und damit werthaltige analytische Inhalte schneller den betrieblichen Entscheidern zur Verfügung gestellt werden.

Weitere Infos zur SAP-SQL-DWH-Testautomation

Weitere Inhalte zur SAP-SQL-DWH-Testautomation, insbesondere auf der Basis des SAP Cloud Application Programming Model, bietet die ISR Information Products AG unter *http://s-prs.de/v781732*.

10.4 Zusammenfassung

In diesem Kapitel sind wir im DevOps-Prozessmodell in den Runtime-Bereich vorgedrungen, in dem es darum geht, die Entwicklungsartefakte im Sinne des Continuous Delivery möglichst zeitnah und automatisiert in die Produktivumgebungen auszuliefern. Die Möglichkeiten, die die SAP-HANA-Plattform mit dem SAP-HANA-XSA-Anwendungsserver bzw. Cloud Foundry hierzu bietet, sind wir dazu Schritt für Schritt durchgegangen. Wir haben unseren Fokus hierzu zunächst auf das manuelle Deployment gerichtet, um die grundsätzliche Struktur des Auslieferungsprozesses nachzuvollziehen. Aus diesem Konstrukt haben wir im Anschluss die Vorgehensweise für ein automatisches Deployment auf der Basis des Git Repositorys und des Build Servers Jenkins erarbeitet. Dieses Szenario haben wir in einem letzten Schritt um das Thema Testautomation ergänzt, bei dem wir Wege aufzeigen, wie die wichtige DevOps-Komponente des Continuous Testing in die Auslieferungskette des SAP SQL DWH integriert werden kann.

Im nächsten und abschließenden Kapitel zum praxisorientierten Teil III des Buches beschäftigen wir uns mit dem Thema Beladung und Betrieb des SAP SQL Data Warehouse.