Quelle: Adobe Stock / putilov_denis

Zukünftige Datenorganisation in ERP-Systemen – eine Analyse etablierter Datenbankansätze

Benedict Bender, Clementine Bertheau, Tim Körppen, Hannah Lauppe und Norbert Gronau

Die digitale Transformation stellt neue Anforderungen an Unternehmenssysteme. Insbesondere ERP-Systeme haben damit zu kämpfen, die neuen Anforderungen aus Industrie 4.0, Multichannel-Handel, Industrial Internet of Things u. a. auf allen Ebenen ihrer Architektur zu erfüllen. Dieser Beitrag zeigt die Notwendigkeit, die Gesamtarchitektur der Systeme zu überdenken und die damit verbundenen Probleme am Ursprung zu adressieren. Viele Einschränkungen der Anpassungsfähigkeit von ERP-Systemen hängen mit der Standardisierung von Daten zusammen. Datenbanken dienen als Grundlage für die Speicherung und den Abruf von Daten, schränken aber die Flexibilität von Unternehmenssystemen und die Möglichkeit, sich an neue Anforderungen anzupassen, ein. Sechs Datenbankkonzepte für ERP-Systeme wurden untersucht, um den ERP-Anbietern aufzuzeigen, wie Datenbankkonzepte zu zukunftsfähigen Lösungen kombiniert verwenden können.

Die digitale Transformation sowie die Weiterentwicklung bestehender Produkte und Dienstleistungen stellen neue Anforderungen Enterprise Resource Planning (ERP)-Systeme. Die Anforderungen ergeben sich aus neuen Bedürfnissen von Organisationen und Märkten sowie aus technologischen Fortschritten. Frühere Studien befassen sich mit der Notwendigkeit, Unternehmenssysteme an neue Anforderungen von Unternehmen anzupassen. Dabei konzentrierten sich frühere Forschungsarbeiten eher auf den Lebenszyklus von Unternehmenssystemen als auf deren architektonischen Aufbau. Bender et al. (2021) identifizierten Probleme mit der aktuellen Generation von ERP-Systemen, die während der ERP-Auswahl, -Implementierung und -Nutzung auftreten und auf das Design und die Architektur des Systems zurückzuführen sind. Um ihre zentrale Rolle beizubehalten, müssen Unternehmenssysteme die kommenden Anforderungen angemessen abdecken.

Die hier dargestellte Untersuchung zeigt, wie wichtig die verschiedenen Architekturschichten für die nächste Generation von ERP-Systemen sind. Die meisten genannten Probleme treten dabei im Zusammenhang mit Datenbanken und Datenmodellen auf. Dabei geht es in erster Linie um den Einsatz von relationalen Datenbanken, da diese heute für die meisten ERP-Anbieter die Datenbanken der Wahl sind. Insofern stellt sich die Frage, inwieweit die vorhandenen Datenbanktypen geeignet sind, zukünftige Geschäftsanforderungen als Grundlage für Unternehmenssysteme zu erfüllen.

Die Inflexibilität von Datenbanken und Datenmodellen zeigt sich zum Beispiel, wenn neue Geschäftsfelder in zentrale ERP-Instanzen integriert werden sollen. Die Integration eines neuen Datenmodells in die bestehende ERP-Datenbank ist sehr kostspielig und führt bei jedem Update des ERP-Systems zu einem zusätzlichen Wartungsaufwand. Ein weiteres Problem ist die Herausforderung bestehender ERP-Systeme, mit Innovationen umzugehen. Dabei kann es sich um ein neues Produkt, einen neuen Prozess handeln. Es ist kompliziert und kostspielig, dem Datenmodell Elemente hinzuzufügen, und es ist fast unmöglich, die Ergänzungen später wieder zu entfernen, wenn sich die Innovation als erfolglos erwiesen hat. Außerdem ist die Integration verschiedener Produkte in ein einziges Datenmodell nur sehr schwer möglich, aber in multinationalen und diversifizierten Unternehmen Realität.

Die Anwendungslandschaften wachsen kontinuierlich, weil durch die zunehmende Digitalisierung größere Wertschöpfungsnetzwerke (inter- und intraorganisatorisch) entstehen. Damit einher gehen immer mehr Daten, auf deren Basis Wertschöpfung betrieben und Entscheidungen getroffen werden sollen. Dies erfordert Ansätze für ERP-Systeme, um Daten aus verschiedenen Systemen, wie z. B. Bestellungen in der Supply Chain,

https://doi.org/10.30844/ERP_22-3_55-62

Kundendaten mit einem CRM-System oder Fertigungsaufträge mit einem Manufacturing Execution System (MES) flexibel zusammenzuführen und zu migrieren.

Die Probleme, die auf der Datenbankebene durch die neuen Anforderungen der digitalen Transformation entstehen, geben Anlass zu der Annahme, dass das relationale Konzept möglicherweise veraltet ist. Um zu erforschen, ob und welche Datenbankeigenschaften einen Engpass für die Bewältigung zukünftiger Geschäftsanforderungen darstellen, wurden verschiedene Datenbanktypen bewertet, um festzustellen, ob sie die zukünftigen Anforderungen an Datenbanken in ERP-Systemen erfüllen.

Datenbanken

Im Folgenden werden sechs konzeptionelle Datenmodelle untersucht. Die Datenbanken wurden ausgewählt, weil sie entweder weit verbreitet oder in der Praxis für andere Zwecke als ERP etabliert sind (Elmasri and Navathe 2007).

Relationale Datenbank
Die relationale Datenbank implementiert das relationale Datenmodell, das typischerweise durch ein Entity-Relationship-Modell modelliert wird (Thalheim 2000). Die Grundlage für das Konzept der relationalen Datenbanken ist eine Relation. Eine Relation ist die mathematische Beschreibung einer Tabelle. Die relationale Algebra definiert die verschiedenen Operationen, die mit der Datentabelle ausgeführt werden können (Elmasri and Navathe 2007). Typischerweise werden die Daten in relationalen Datenbanken in Form von mehreren Tabellen dargestellt, die zueinander in Beziehung gesetzt werden. Die Eigenschaften der einzelnen Tabellen und die Beziehungen zwischen den Tabellen sind durch das Schema der relationalen Datenbank vordefiniert. Datenbanksysteme, die relationale Datenbanken implementieren, werden als relationale Datenbankmanagementsysteme (RDBMS) bezeichnet. Der relationale Datenbankansatz wurde in den 1970er Jahren entwickelt. Relationale Datenbanksysteme werden in der Regel mit der Structured Query Language (SQL) abgefragt. In den späten 2000er Jahren hat sich ein neuer Ansatz für skalierbarere relationale Datenbanken entwickelt. Diese so genannten NewSQL-Datenbanken folgen einer relationalen Struktur und Semantik und bieten gleichzeitig eine höhere Skalierbarkeit und Konsistenz (als relationale oder NoSQL-Datenbanken). Dazu trägt vor allem die Partitionierung der Daten in kleinere Bereiche ("Shards") bei. Zudem ermöglicht dies verteilte Architekturen, die in Zeiten von BigData und Cloud-Computing deutlich an Bedeutung gewonnen haben.

Spaltenbasierte Datenbank
Spaltenorientierte Datenbankmanagementsysteme (CDBMS) speichern Daten spaltenweise und nicht zeilenweise (wie es bei relationalen Datenbanken üblich ist). Spaltenorientierte Datenbanken unterscheiden sich daher in der Datenspeicherung und der physischen Darstellung der Daten, was zu unterschiedlichen Möglichkeiten bei Anfragen an die Daten führt (Elmasri and Navathe 2007). Während zeilenorientierte Datenbanken besser für Online-Transaktionsverarbeitungssysteme (OLTP) geeignet sind, eignen sich spaltenorientierte Datenbanken besser für analytische Aufgaben wie Online-Analytical-Processing-Systeme (OLAP) (Abadi et al. 2013). Für die Berichterstattung werden in der Regel nicht alle Attribute einer Zeile benötigt, so dass der Zugriff auf eine einstellige Anzahl von Spalten viel schneller ist als das herkömmliche Scannen von Zeilen nach diesen Attributen (Abadi et al. 2009).

Objekt-relationale Datenbank
Objektorientierte Datenbanken werden für die Speicherung von Informationen über Objekte mit komplexen Strukturen verwendet (Atkinson et al. 1990). Objektorientierte Datenbanken erlauben es, individuelle Eigenschaften von Objekten innerhalb von Softwareanwendungen abzudecken. Sie integrieren strukturelle Vorgaben und die auf diese Objekte anwendbaren Operationen, wobei sich verschiedene Objekte typischerweise in ihren Möglichkeiten unterscheiden. Objekte, Klassen und Vererbung werden direkt im Datenbankschema und in der Sprache dargestellt. Diese Klasse von Datenbanken ist für die objektorientierte Programmierung konzipiert oder kann direkt von ihr abgeleitet werden (Elmasri and Navathe 2007). Es besteht keine Notwendigkeit, verwandte Datenstrukturen separat zu modellieren. Der Ansatz kombiniert die Spezifikationen während der Anwendungskonzeption und -programmierung. Die Anwendungsentwicklung und die Datenbankspezifikationen werden bei diesem Ansatz integriert. Ähnlich wie bei relationalen Datenbanken werden die Daten in Tabellen gespeichert und das Ergebnis einer (SQL-)Abfrage ist ebenfalls eine Tabelle (Mahnke and Steiert 2000).

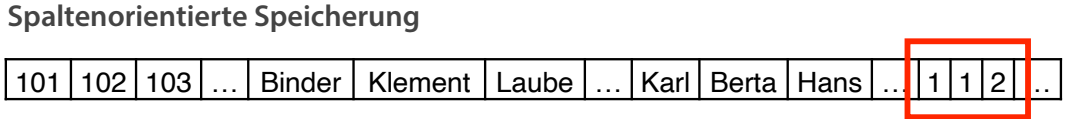

Bild 1: Ansatzpunkte zur Umsetzung [6].

Graph-Datenbank
Graphdatenbanken gehören zur Familie der "nicht nur"-SQL-Datenbanken (NoSQL) (Benymol and Abraham 2020). Der Hauptunterschied zu traditionellen Konzepten ist die Möglichkeit, ein starres Schema zu vermeiden. Daher lassen sich verschiedene (moderne) Datenbankkonzepte der NoSQL-Familie zuordnen. Dies gilt auch für Graphdatenbanken. Generell basieren Graphdatenbanken auf graphentheoretischen Konzepten, d. h. der Darstellung von Inhalt und Beziehung von Datensätzen in Knoten und Kanten (und Eigenschaften, je nach Implementierung). Daher würden objektorientierte Anwendungen aufgrund eines flexibleren Schemas besser zu Graphdatenbanken passen (Chen et al. 2020). Die gebräuchlichste Darstellung ist der Labeled Property Graph (LPG). Er besteht aus Knoten, die die Werte/Eigenschaften enthalten und mit Beschriftungen versehen werden können (z. B. zur Gruppierung), sowie aus gerichteten, beschrifteten Kanten, die Beziehungen zwischen den Knoten darstellen (Robinson et al. 2015). Diese Beschriftungen für die Knoten und Kanten stellen Metadaten für Graphalgorithmen dar und verbessern die Lesbarkeit und Semantik.

Key-Value-Datenbank
Ähnlich wie Netzwerk- und Graphdatenbanken sind Key-Value-Datenbanken mit der NoSQL-Familie verwandt. Im Allgemeinen bilden Key-Value-Datenbanken Sätze von Schlüsseln (d. h. einen individuellen und eindeutigen Bezeichner) auf entsprechende Werte (d. h. den eigentlichen Inhalt des Datensatzes) ab. Neben dieser Zuordnung gibt es keine Struktur oder Beziehung in der Organisation der Daten selbst. Daher können Key-Value-Stores auch als schemalos bezeichnet werden (Sadalage and Fowler 2013). Die Manipulation der Daten ist auf einfache CRUD-Operationen (Erstellen, Lesen, Aktualisieren, Löschen) beschränkt. Diese Beschränkungen bei der Organisation und Manipulation sind gleichzeitig die Hauptvorteile von Key-Value-Datenbanken: Die Einfachheit führt zu geringen Latenzzeiten und hohem Durchsatz (Gessert et al. 2017).

Dokumentenbasierte Datenbank
Reine Dokumentenspeicher sind eine Erweiterung von Key-Value-Stores und folgen daher NoSQL-Eigenschaften. Jeder Datensatz (Wert) entspricht einem eindeutigen Schlüssel, mit dem Abfragen und Manipulationen durchgeführt werden. Der Hauptunterschied besteht darin, dass die Werte auf semistrukturierte Formate (z. B. JavaScript Object Notation oder Extensible Markup Language) beschränkt sind. Bei der Abfrage von Daten aus dokumentenbasierten Datenbanken ist es möglich, ganze Dokumente auszuwählen oder Teile des Inhalts von Dokumenten anzugeben (Gessert et al. 2017). Darüber hinaus erlaubt die Flexibilität der Dokumentenstruktur die Änderung von Attributen (z. B. Hinzufügen oder Entfernen von Attributen) zur Laufzeit (Davoudian et al. 2018).

Datenbank-Anforderungen an zukünftige ERP-Systeme
Bei der Softwareentwicklung und dem Entwurf von Informationssystemen ist die Bewertung der relevanten Anforderungen von größter Bedeutung. Sie gewährleistet eine präzise Entwicklung und später die Eignung für den realen Einsatz. Bei Unternehmenssystemen, insbesondere bei ERP-Systemen, können diese Anforderungen alle Schichten und Komponenten der Systemarchitektur betreffen. Da die Hauptaufgaben von ERP-Systemen die Verwaltung von Ressourcen und Geschäftsobjekten sind, sind die Anforderungen an die Datenbanken von ERP-Systemen von großer Bedeutung für die Prozessausführung und die (Daten- und Prozess-)Konsistenz.

Relevante Anforderungen für die Datenbankebene sind in Tabelle 1 dargestellt.

Fokus auf Geschäftsprozesse
Das Thema der Geschäftsprozessorientierung beschreibt Anforderungen, die sich direkt aus den Geschäftsprozessen und deren technologischen Artefakten ableiten. Für (ERP-)Datenbanken hat die Transformation von Prozessen Konsequenzen in der Datenhaltung, insbesondere der Skalierbarkeit im Hinblick auf Datenlasten. So ist z. B. in Geschäftsmodellen mit vorherrschender Kundenorientierung oder im Bereich des (industriellen) Internet der Dinge eine effiziente Integration und Transformation großer Datenmengen bei gleichzeitiger Skalierbarkeit in Bezug auf die aktuellen Daten / Workloads erforderlich (ur Rehman et al. 2019). Neben der Skalierbarkeit des Datenvolumens steht die Interaktionseffizienz der Datenbank in einer wechselseitigen Beziehung zur Prozesseffizienz. Geschäftsmodelle und Unternehmensfunktionen, die hochfrequente Transaktionen (z. B. in Produktionsumgebungen oder Kundeninteraktion), Transformation und Verarbeitung (sowohl in transaktionaler als auch analytischer Ausrichtung) in der Datenbank vorsehen, müssen nahezu in Echtzeit möglich sein. Für die Effizienz von Transaktionen sind sowohl Änderungen an einzelnen Datensätzen als auch Stapeländerungen erforderlich. Während die Effizienz der Datenverarbeitung im Vordergrund steht, muss die Konsistenz, Robustheit und Fehlerfreiheit dieser Transaktionen gewährleistet sein. Hier kann die Abbildung von Transaktionen und Ergebnissen von Geschäftsprozessen in der Datenverwaltung durch die Einführung von Semantik implizit die Leistung verbessern. Die strukturanaloge Abbildung von realen Geschäftsprozessen und Geschäftsobjekten in Datenbanken (z. B. der digitale Zwilling von Produkten oder Anlagen) ist entscheidend für eine robuste und effiziente Planung und Ausführung von Aufgaben (Ma et al. 2019). Gleichzeitig entstehen bei der individuellen Massenproduktion (Mass Customization) große Mengen von untereinander unterschiedlichen Objekten (Tupelgröße eins, d. h. Singleton-Objekte), die in den aktuellen Datenmodellen von ERP-Systemen nicht effizient dargestellt werden können.

https://doi.org/10.30844/ERP_22-3_55-62

Kategorie	Annahme für ERP	Operationalisierung für ERP
2.2.1 Fokus auf Geschäftsprozesse	Anforderungen, die sich direkt aus den Geschäftsprozessen ergeben	Skalierbarkeit Leistung (OLAP) Leistung (OLTP) – Änderung eines Datensatzes und Durchführung einer Aufgabe über viele Datensätze hinweg Echtzeitfähigkeit Digitaler Zwilling Heterogenität der Geschäftsobjekte
2.2.2 Schwerpunkt Integration	Anforderungen, die die Integration bestehender Anwendungen betreffen.	ETL Prozess-Schema Interoperabilität
2.2.3 Verteilte Anforderungen	Anforderungen verschiedener Interessengruppen als auch solche, die aufgrund organisatorischer und geografischer Verteilung im Geschäftsbetrieb	Abbildung des Datenschutzes im Datenmodell Abbildung der Compliance-Anforderungen im Datenmodell Abbildung der Kooperationseignung im Datenmodell Abbildung der Datensicherheit in Bezug auf die Beteiligten im Datenmodell
2.2.4 Schichten von Anforderungen	Es werden Anforderungen behandelt, die verschiedene Abstraktionsebenen betreffen.	Granulare Sicherheitseinstellungen Verständlichkeit des Datenmodells
2.2.5 Zentrale Bedeutung der Architektur	Anforderungen, die sich auf architektonische Gegebenheiten beziehen und damit indirekt die Anwendung beeinflussen, werden unter dem Thema Architekturzentralität zusammengefasst.	Modularität Abbildung der Prozessstruktur durch Datenmanagement
2.2.6 Interdependente Komplexität	Die Anforderungen an die Komplexität von Softwaresystemen sind erheblich gestiegen.	Vermeidung von zusätzlicher Komplexität Strukturbildung für unstrukturierte Daten Unvollständige Daten

Tabelle 1: Anforderungen und Operationalisierung

Schwerpunkt Integration
Die Anforderungen für das Thema Integration konzentrieren sich auf die Integration der bestehenden Anwendungen. Dies beinhaltet sowohl die Anforderung der Interoperabilität der Systeme, d. h. die Anpassungsfähigkeit an unterschiedliche Betriebsarten der Datenbanken (Datenbanksysteme), Standards und Schnittstellen als auch die Interoperabilität der Datenbankschemata. Dabei muss der eigentliche Inhalt der Datenbank und deren Organisation durch die Schemata für eine reibungslose Kombination und gleichzeitige Nutzung anpassbar sein. Neben der Interoperabilität von mehreren Datenbanken und Datenbankschemata ist auch die Interoperabilität von Datenbanken und übergeordneten Informationssystemen erforderlich. Daher sind Verbesserungen der Extraktions-, Transformations- und Ladeprozesse (insbesondere eine verbesserte und automatisierte Extraktion und Transformation von Daten) erforderlich, um die Effizienz bei der Verarbeitung großer Datenmengen und beim Umgang mit heterogenen Datentypen zu steigern.

Verteilte Anforderungen
Zu den verteilten Anforderungen gehören sowohl die Anforderungen verschiedener Interessengruppen als auch solche, die aufgrund organisatorischer und geografischer Entfernungen im Geschäftsbetrieb verteilt sind. In Bezug auf (ERP-)Datenbanken sind verteilte Anforderungen aus Gründen der Zusammenarbeit, der Sicherheit und des Datenschutzes relevant. Die weitere Automatisierung von Geschäftsprozessen erfordert, dass Datenbanken selbständig über die betreffenden Datenelemente kommunizieren (z. B. ein Datenobjekt ist für Kooperationszwecke vorgesehen). Darüber hinaus sollte das Datenmodell Einstellungen für Compliance, Datenschutz und Schutz/Sicherheit für die zugrunde liegenden Daten enthalten.

Schichten von Anforderungen

Bei Anforderungsschichten werden verschiedene Abstraktionsebenen durch die Anforderung angesprochen. Mit zunehmender Komplexität und Größe von Datenmodellen ist die Unterstützung der menschlichen Lesbarkeit für verschiedene Schichten der Datenbank sowie deren inhaltliche und strukturelle Darstellung erforderlich. Dies könnte zum Beispiel den Kontext, die Beziehungen und den Inhalt der dargestellten Daten umfassen. Während dies zu einer verbesserten Lesbarkeit und Verständlichkeit führt, entstehen Anforderungen an eine granularere Ebene der Sicherheit und Rechteverwaltung – vom übergeordneten Datenbankschutz bis hin zu tabellenbasierten Selektionen.

Zentrale Bedeutung der Architektur
Anforderungen, die auf architektonische Gegebenheiten abzielen und damit indirekt die Anwendung beeinflussen, werden unter dem Thema Architekturzentralität zusammengefasst. Während modulare Software Stand der Technik ist, werden die Vorteile von Modularität (insbesondere höhere Flexibilität, Zuverlässigkeit, Verfügbarkeit, Skalierbarkeit und geringere Kosten) bei Datenbanken

und -modellen meist noch nicht genutzt. Es gibt zwar verschiedene Ansätze für modulare Datenbankarchitekturen und verteilte Datenbanken (Irmert et al. 2008; Parent et al. 2009; Seybold and Domaschka 2017)die aber in der praktischen Anwendung in Unternehmenssystemen kaum zu finden sind. Neben den bereits erwähnten Vorteilen der Modularität könnten modulare Datenbanken in ERP-Systemen dazu beitragen, dass die Struktur übergeordneter Geschäftsprozesse (Prozessanalogie) aktiv im Datenmodell abgebildet wird.

Interdependente Komplexität

Im Allgemeinen ist die Komplexität von Softwaresystemen erheblich gestiegen. Daher werden Anforderungen, die die Komplexität von Anforderungen (insbesondere Interdependenzen) adressieren, in der Thematik der interdependenten Komplexität gefördert. Sowohl detaillierte als auch individualisierte Darstellungen von Geschäftsobjekten und -prozessen erhöhen die Komplexität. Datenbanken sind gefordert, genügend Details des realen Gegenstücks abzubilden und dennoch die entstehende Komplexität im Datenmodell zu beherrschen. Hier ergeben sich Anforderungen an die aktive Vermeidung weiterer (unnötiger) Komplexität durch die Datenbank(systeme). Während Komplexität vermieden werden soll, produzieren neu verfügbare (analoge) Datenquellen halb-, unstrukturierte und unvollständige Daten (z. B. Sprachaufzeichnungen oder Bilder) und erzeugen im Gegenzug Komplexität. Diese halb-, unstrukturierten und unvollständigen Datensätze werden zunehmend relevant für die Wertschöpfung. Daher müssen Datenbanken die (effiziente) Integration, Transformation und Speicherung dieser Datenformen ermöglichen.

Fließendes Design

Bislang bezogen sich die Anforderungen auf die Gestaltung der Datenbank. Im Gegensatz dazu befasst sich das Thema "Fluidität des Designs" mit Anforderungen, die die Weiterentwicklung der Datenbank, d. h. nach der Implementierung, betreffen. Dies ist von besonderer Bedeutung, da ERP-Systeme oft über einen längeren Zeitraum betrieben werden. Bei Datenbanken können sich im Betrieb Anforderungen an Änderungen des Datenbanksystems sowie der (strukturellen) Repräsentation des Datenmodells ergeben. Ersteres erfordert Anpassungsfähigkeit und Wartbarkeit durch Änderungen des Datenbanksystems zur Laufzeit, z. B. von Schnittstellen. Bei letzterem geht es um Änderungen des Schemas, d. h. der inhaltlichen Struktur der Datenbank. Hier ergeben sich Anforderungen aus der Entwicklung der Geschäftsprozesse und -abläufe, die Einfluss auf die Darstellung des Datenmodells haben. Generell werden einfache Änderungen zur Laufzeit benötigt, um teure Entwicklungszyklen und Ausfallzeiten zu verringern. Außerdem müssen Datenbanken aktiv den Optimierungsaufwand senken, um die Änderungs- und Benutzerfreundlichkeit zu verbessern.

Schlussfolgerungen

Da relationale und spaltenbasierte Datenbanken eher statische Datenbankschemata aufweisen und daher ähnliche Merkmale in Bezug auf die Datenorganisation aufweisen, werden diese beiden Datenbanken im Folgenden als tabellarische Datenbanken subsumiert. Objektrelationale, Graph- und Dokumentdatenbanken bieten flexiblere, objektorientierte Schemata. Diese Datenbanktypen werden daher als strukturell-analoge Datenbanken bezeichnet. Schema-freie Key-Value-Datenbanken schließlich bilden eine eigene Kategorie, da sie keine strukturellen Ähnlichkeiten mit den anderen Typen aufweisen.

Wie bereits erwähnt, sind tabellarische Datenbanken im Bereich der ERP-Systeme weit verbreitet (Plattner 2009). Neben den Problemen, die mit tabellarischen Datenbanken für ERP-Systeme verbunden sind, wurden in der zuvor vorgestellten Analyse Nachteile und Vorteile tabellarischer Datenbanken aufgezeigt. Generell bieten tabellarische Datenbanken im Hinblick auf das Thema Integrationsfokus Vorteile gegenüber strukturanalogen Datenbanken. Die statischen, normalisierten Schemata von tabellarischen Datenbanken ermöglichen eine nahtlose Integration von ETL-Prozessen und Datenbankschemata. Das Thema der verteilten Anforderungen (d. h. die direkte und flexible Erweiterung des Datenmodells und der modellierten Geschäftsobjekte um Metadaten) wird dagegen von schema-analogen Datenbanken weitgehend adressiert. Rein tabellarische Darstellungen verhindern die Objektorientierung und die granulare Verarbeitung von Datenobjekten (siehe auch das Thema Schichten von Anforderungen). Diese Flexibilität und Objektorientierung erlaubt es strukturell-analogen Datenbanken zudem, Merkmale der Fluidität des Designs, d. h. der kontinuierlichen Entwicklung und Veränderung von Datenstrukturen, zu übernehmen. Dies wird auch bei den Architekturanforderungen im Bereich der Zentralität der Architektur beobachtet. Modularität und Strukturanalogie werden vor allem durch strukturanaloge Datenbanken angesprochen. Gerade in Bezug auf letztere werden tabellarische Datenbanken den Anforderungen nicht gerecht. Dies führt teilweise auch zu weniger nachvollziehbaren Datenmodellen (Anforderungsschichten) im Vergleich zu strukturanalogen Datenbanken. Für das Thema Geschäftsprozessorientierung hat die Analyse keine generellen, entscheidenden Vor- oder Nachteile zwischen tabellarischen und strukturanalogen Datenbanken ergeben. Da jedoch die Verwaltung von Geschäftsprozessen die primäre Aufgabe von ERP-Systemen ist, ist die Geschäftsprozessorientierung für zukünftige Datenbankmodelle von besonderer Bedeutung.

Insgesamt ergab die Analyse der vorab bewerteten Anforderungen Stärken und Schwächen für alle Datenbankmodelle (Bild 2). Traditionelle, tabellarische Datenbanken bieten Vorteile für die Speicherung und Verarbeitung ähnlicher und wiederkehrender Datensätze von Transaktionsdaten (z. B. Prozessdaten, Abrechnungsdaten, Sensordaten). Strukturell-analoge Datenbanken hingegen

https://doi.org/10.30844/ERP_22-3_55-62

zeichnen sich durch die Abbildung individueller, flexibler und sich verändernder Geschäftsobjekte aus. Die erhöhte Flexibilität von NoSQL-Datenbanken ermöglicht eine bessere Abbildung von Geschäftsobjekten, während sie mit Schwierigkeiten hinsichtlich der Transaktionskonsistenz (ACID; atomic, consistency, isolation and durability) zu kämpfen haben (Radulović et al. 2016; Ekren and Erkollar 2020; Sokolova et al. 2020). Die Antwort auf die eingangs gestellte Frage lautet also, dass es nicht nur ein einziges Datenbankmodell gibt, das für die Erfüllung zukünftiger Datenbankanforderungen geeignet ist.

Ein Multimodell-Datenbankansatz für Unternehmenssysteme

Um die bestehenden Beschränkungen von Datenbanken im Zusammenhang mit Unternehmenssystemen zu überwinden, wurden Ansätze für Multi-Modell-Datenbanken entwickelt, die es ermöglichen, die Vorteile mehrerer Datenbanksysteme zu kombinieren. Dementsprechend kann ihre individuelle Spezialisierung genutzt werden, um zukünftige Geschäftsanforderungen im Bereich der Unternehmenssysteme zu erfüllen. Die Idee von Multi-Modell-Datenbanken besteht darin, mehrere Datenmodelle in einer integrierten Umgebung zu unterstützen. Im Gegensatz zu dem Ansatz, sich nur auf ein einziges Datenmodell zu stützen, ermöglicht dies eine flexiblere Anpassung.

Als Lösung könnten hybride Datenbankarchitekturen die zukünftigen Anforderungen an ERP-Systeme am besten erfüllen. Durch die Kombination von tabellarischen (SQL-)Datenbanken und schema-analogen (meist NoSQL-)Datenbanken in einer hybriden Datenbankarchitektur werden die Vorteile beider Datenbankmodelle synergetisch genutzt (Nimis et al. 2014; Radulović et al. 2016; Sokolova et al. 2020). Einige NoSQL-Datenbanksysteme (z. B. MongoDB) bieten dedizierte Funktionalität für die Zusammenarbeit mit relationalen Datenbanken und sind daher (technisch) für diesen Einsatz geeignet (Radulović

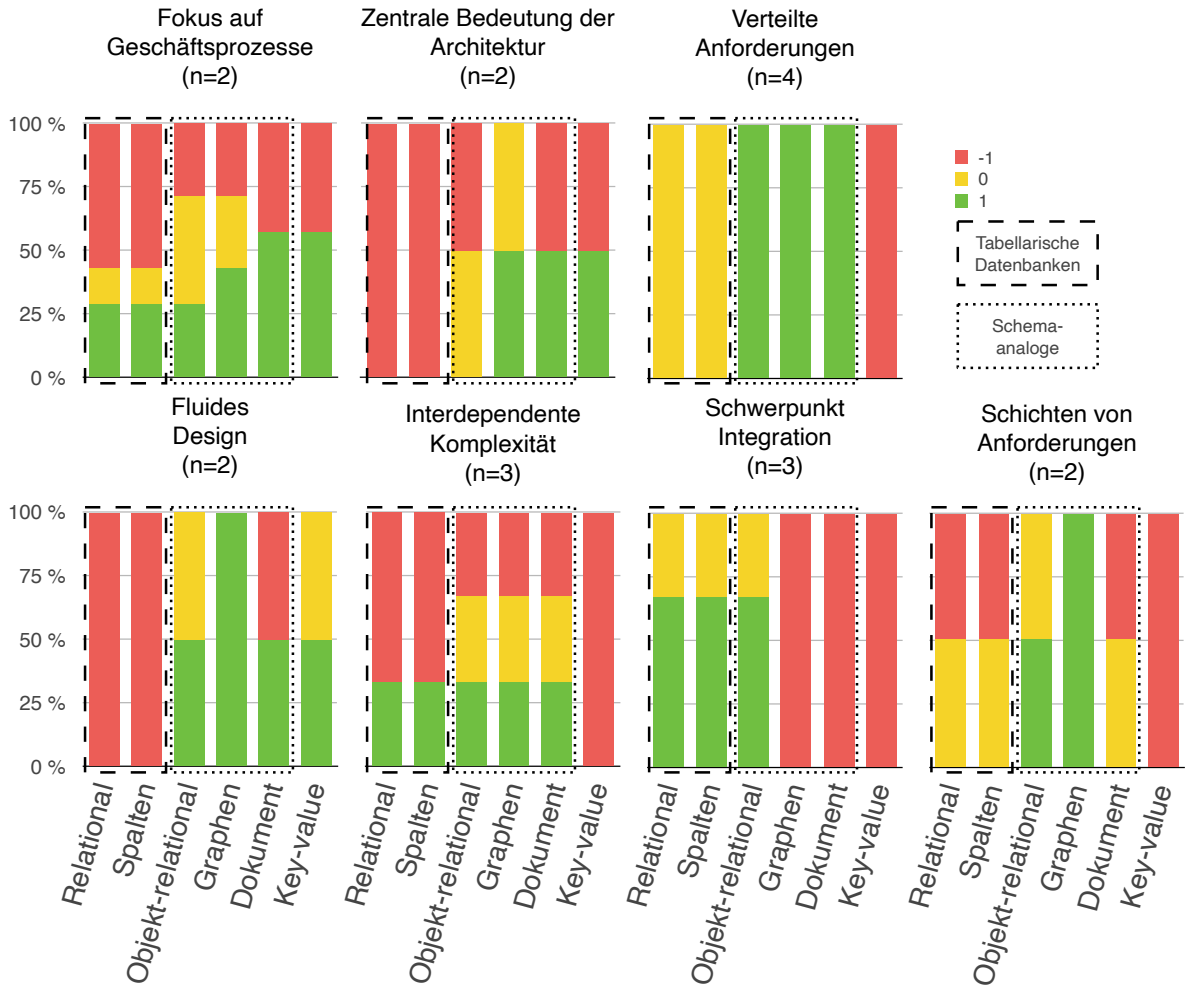

Bild 2: Erfüllung der ERP-Anforderungen nach Datenbanktyp.

et al. 2016). In der bisherigen Forschung wurde bereits die Idee eines hybriden (SQL und NoSQL) Datenbankmodells vorgestellt, das auf der Strukturiertheit der Daten basiert (Bjeladinovic 2018). Für ERP-Systeme erweitert unsere Analyse und der daraus resultierende Vorschlag diesen Ansatz (Bild 3): Die Trennung von Daten auf Basis der Strukturiertheit ist nicht nur für Unternehmenssysteme geeignet, sondern könnte auch um die Unterscheidung zwischen Stammdaten (a) und Transaktionsdaten (b) erweitert werden. Stammdaten von Geschäftsobjekten (z. B. Ressourcen, Produkte, Humankapital, Kunden, Akteure der Lieferkette) müssen viele verschiedene Merkmale und Verhaltensweisen (Prozesse) von Geschäftsobjekten erfassen. Diese Unterschiede zwischen Geschäftsobjekten könnten in flexiblen strukturanalogen Datenbanken gespeichert werden. Der Umgang mit vielen verschiedenen Varianten (was in relationalen Datenbanken schnell zu hoher Komplexität führt) kann durch schema-analoge Datenbankmodelle leicht realisiert werden. Transaktionsdaten (mit hohen Anforderungen an die ACID-Konformität) würden weiterhin in relationalen Datenbanken gespeichert werden (z. B. Aufträge, Rechnungen, Zahlungen, Lagersätze, Fahrpläne). Diese Trennung nutzt die Vorteile der Schemaflexibilität in schemaanalogen Datenbanken für die Modellierung von Geschäftsobjekten und gewährleistet gleichzeitig die Konsistenz der Transaktionsdaten in tabellarischen Datenbanken. Für die Integration und Orchestrierung dieser (tabellarischen und strukturanalogen) Datenbanken sind spezielle Verfahren zur Synchronisation, insbesondere im Hinblick auf die Einheitlichkeit der Datensätze, notwendig. Dies könnte z. B. durch Virtualisierungsalgorithmen erfolgen, die eine gemeinsame (SQL-)Abfrage sowohl der tabellarischen als auch der strukturanalogen Datenbank ermöglichen (Lawrence 2014). Darüber hinaus wurden in der Forschung auch Konzepte für die Schematransformation und Abfrageübersetzung zwischen SQL- und NoSQL-Datenbanken vorgeschlagen (Dai 2019).

Darüber hinaus wird dem zunehmenden Bedarf an Leistung und Verteilung durch Key-Value-Datenbanken begegnet, die das Zwischenspeichern von hochfrequenten Prozessdaten ermöglichen, die kurz nach ihrer Verarbeitung verworfen werden können ("Puffer-Datenbanken") (Nimis et al. 2014). Diese Trennung zwischen Datenbanken auf der Grundlage der Struktur und des Zwecks der Daten (Stamm- und Transaktionsdaten) würde eine effiziente und robuste Ausführung von Geschäftsprozessen unter Berücksichtigung der individuellen Merkmale der verarbeiteten Geschäftsobjekte ermöglichen. Solche hybriden Architekturen von SQL- und NoSQL-Datenbanken sind in der Literatur bereits behandelt worden (Nance et al. 2013; Bjeladinovic 2018; Bjeladinovic et al. 2020). Obwohl diese Ansätze für Unternehmenssysteme konzipiert wurden, gab es keinen direkten Fokus auf ERP-Systeme. Nichtsdestotrotz sind die in der Literatur identifizierten Vorteile ähnlich wie die in dieser Arbeit vorgeschlagenen Optionen für ERP: Insbesondere in Bezug auf Leistung, Skalierbarkeit und Heterogenität sollen NoSQL-Datenbanken die Schwächen traditioneller SQL-Datenbanken kompensieren (Nance et al. 2013). Die in diesem Beitrag vorgeschlagene Trennung zwischen Stamm- und Bewegungsdaten erweitert diese Ansätze der Literatur und passt sie an ERP-Systeme an, indem sowohl die Trennung auf Strukturiertheit (Bjeladinovic 2018) sondern auch nach dem Zweck der (Unternehmens-)Daten.

Da ERP-Systeme in der Regel unternehmensweit eingesetzt werden, wird von ihnen erwartet, dass sie eine Vielzahl unterschiedlicher Geschäftsobjekte verwalten. Diese Arbeit konzentriert sich auf die Erfüllung der Anforderungen an die Speicherstruktur verschiedener Datenbanken. Dabei wurden spezifische Implementierungen (Datenbankmanagementsysteme; DBMS) bewusst außer Acht gelassen, da diese Datenbankmanagementsysteme in der Regel nur die umgebende Funktionalität (z. B. Schnittstellen und Abfragesprachen) ergänzen. Die zugrundeliegende Struktur der Datenorganisation lässt sich nach wie vor einem der in dieser Arbeit diskutierten allgemeinen Datenbanktypen zuordnen. Neben den Anforderungen an die Datenorganisation innerhalb einer ERP-Datenbank ergaben sich im Rahmen der Erstbewertung jedoch auch allgemeinere Anforderungen an das DBMS. Hierbei handelt es sich um Anforderungen zum Thema interdependente Komplexität, d. h. Vermeidung von Komplexität und effizienter Umgang mit unvollständigen und unstrukturierten Daten. Mit zunehmender Heterogenität der Geschäftsobjekte und Konzepten wie Big Data dürften diese Anforderungen in Zukunft noch mehr an Relevanz gewinnen. DBMS (unabhängig vom zugrundeliegenden Datenmodell) sind gefordert, aktiv Komplexität zu vermeiden und den Optimierungsaufwand zu reduzieren. Bei hybriden Datenbankmodellen (wie in diesem Beitrag vorgeschlagen) müssen diese DBMS zudem die Robustheit von Transaktionen und die Verarbeitung von Daten zwischen (verschiedenen) Datenbankmodellen der hybriden Architektur und höheren Softwareschichten sicherstellen. Besonderes Augenmerk sollte dabei auf Aspekte der Stammdatenqualität gelegt werden. Bereits in (traditionellen) relationalen Datenbanken erfordert die Pflege der Stammdaten einen erheblichen Zeit- und Arbeitsaufwand (Knolmayer and Röthlin 2006; Haug et al. 2013).

Ausblick

Insgesamt könnten hybride (multi-model) Architekturen, die tabellarische Datenbankmodelle wie relationale Datenbanken mit strukturanalogen Datenbanken wie Graphen- oder objektrelationalen Datenbanken kombinieren, durch Kombination der individuellen Vorteile dieser Datenbanken zukünftige Anforderungen erfüllen. Dies bringt sowohl für ERP-Systementwickler als auch für Entwickler von Datenbanksystemen Veränderungen mit sich. Einerseits müssen sich die Entwickler von Datenbanksystemen auf modularere, föderierte und interoperable Architekturen einstellen, indem sie die

https://doi.org/10.30844/ERP_22-3_55-62

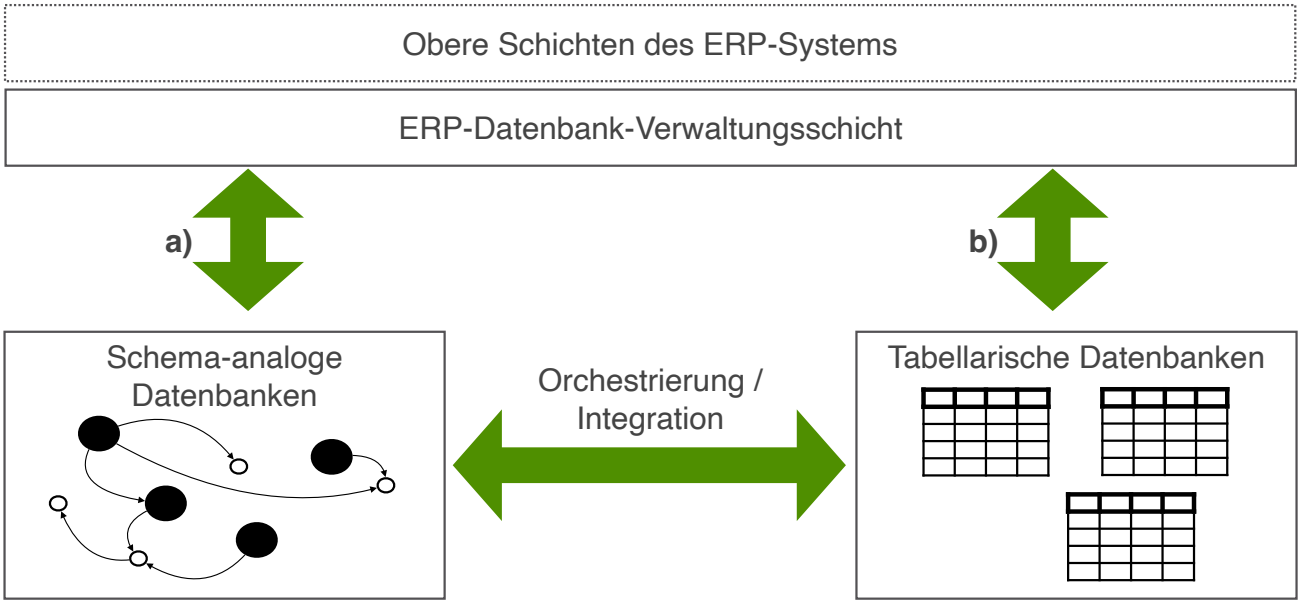

Bild 3: Abgrenzung zwischen schemaanalogen und tabellarischen Datenbanken.

zuvor genannten Anforderungen an das Datenbankmanagementsystem berücksichtigen. Andererseits sind die Entwickler von ERP-Systemen gezwungen, verschiedene Implementierungen unterschiedlicher Datenbanksysteme zu evaluieren und sie entsprechend den vorhandenen Softwarekomponenten zu integrieren. Die Ergebnisse können nur dazu dienen, die bestehende Architektur von Unternehmenssystemen hinsichtlich der Datenbankschicht zu überdenken. Die Ergebnisse ermöglichen es den Systemanbietern, sich an neue Geschäftsanforderungen anzupassen. Dabei sind Systemanbieter gut beraten, sich nicht nur auf Standardprodukte für ihre Systeme zu verlassen, sondern mehrere Ansätze zu kombinieren oder Multi-Modell-Datenbanken als Grundlage für zukünftige Systemgenerationen zu betrachten.

Dieser Beitrag stellt die gekürzte Fassung eines englischsprachigen Forschungsbeitrags dar, der unter der DOI 10.1007/s10257-022-00555-6 abrufbar ist. Dort finden sich auch die in diesem Beitrag genannten Quellen.

Schlüsselwörter:
Datenbank, Unternehmenssystem, ERP-System, Anforderungen, Probleme, Zukunft

The digital transformation places new demands on enterprise systems.

ERP systems in particular, as the predominant class of enterprise systems, are struggling to meet the new requirements from Industry 4.0, multichannel commerce, Industrial Internet-of-things and others at all levels of their architecture. This paper demonstrates the urgent need to rethink the overall architecture of systems and get to the root of the associated problems. Many limitations to the adaptability of ERP systems are related to the standardization of data. Databases serve as the foundation for storing and retrieving data, but they limit the flexibility of enterprise systems and the ability to adapt to new requirements. Six database concepts for ERP systems were examined to suggest ERP vendors use other database concepts in the future.

Keywords:
Database, Enterprise system, ERP System, requirements, problems, future

Dr. Benedict Bender leitet die Forschungsgruppe Plattformen und Softwareplattformen an der Universität Potsdam.

Clementine Bertheau ist Consultant bei der Cassini AG, Berlin.

Tim Körppen und **Hannah Lauppe** sind Mitglieder der Forschungsgruppe Plattformen und Softwareplattformen.

Univ.-Prof. Dr.-Ing. habil. Norbert Gronau ist Inhaber des Lehrstuhls für Prozesse und Systeme an der Universität Potsdam.