1.3 Merkmale eines DBMS

1.3.1 Aufgaben eines DBMS

Gemäß seiner Definition muss ein DBMS folgende Funktionalitäten bieten:

Integrierte Datenhaltung. Das DBMS ermöglicht die *einheitliche* Verwaltung *aller* von den Anwendungen benötigten Daten. Somit wird jedes logische Datenelement, wie beispielsweise der Name eines Kunden, *an nur einer* Stelle in der Datenbank gespeichert. Dabei muss ein DBMS die Möglichkeit bieten, eine Vielzahl komplexer Beziehungen zwischen den Daten zu definieren sowie zusammenhängende Daten schnell und effizient miteinander zu verknüpfen. In manchen Fällen kann eine *kontrollierte Redundanz* allerdings nützlich sein, um die Effizienz der Verarbeitung von Anfragen zu verbessern (→ 9.3.1.3).

Sprache. Das DBMS stellt an seiner Schnittstelle eine Datenbanksprache (*query language*) für die folgenden Zwecke zur Verfügung:

- Datenanfrage (retrieval),
- Datenmanipulation (Data Manipulation Language, DML),
- Verwaltung der Datenbank (Data Definition Language, DDL),
- Berechtigungssteuerung (Data Control Language, DCL).

Da viele verschiedene Benutzer mit unterschiedlichen technischen Kenntnissen und Anforderungen auf eine Datenbank zugreifen, sollte ein DBMS eine Vielzahl von Benutzeroberflächen bereitstellen. Dazu zählen Anfragesprachen für gelegentliche Benutzer, Programmierschnittstellen und grafische Benutzeroberflächen (*Graphical User Interface, GUI*). Die Möglichkeit eines Webzugriffs ist heute eine Grundvoraussetzung für den Einsatz eines DBMS.

Katalog. Der Katalog (*data dictionary*) ermöglicht Zugriffe auf die Datenbeschreibungen der Datenbank, die auch als Metadaten bezeichnet werden.

❑ *Beispiele:* Eigentümer und Erzeugungszeit von Datenbankobjekten (Tabellen), Beschreibung von Tabellen und ihren Spalten (Datentyp, Länge).

Benutzersichten. Für unterschiedliche Klassen von Benutzern sind verschiedene Sichten (*views*) erforderlich, die bestimmte Ausschnitte aus dem Datenbestand beinhalten oder diesen in einer für die jeweilige Anwendung benötigten Weise strukturieren. Die Sichten sind im externen Schema der Datenbank definiert (→ 1.4.2.2).

Konsistenzkontrolle. Die Konsistenzkontrolle, auch als *Integritätssicherung* bezeichnet, übernimmt die Gewährleistung der Korrektheit von Datenbankinhalten und der korrekten Ausführung von Änderungen. Ein korrek-

ter Datenbankzustand wird durch benutzerdefinierte Integritätsbedingungen (*constraints*) im DBMS definiert, die während der Laufzeit der Anwendungen vom System kontrolliert werden. Daneben wird aber auch die physische Integrität sichergestellt, d. h. die Gewährleistung intakter Speicherstrukturen und Inhalte (Speicherkonsistenz).

Datenzugriffskontrolle. Durch die Festlegung von Regeln kann der unautorisierte Zugriff auf die in der Datenbank gespeicherten Daten verhindert werden (*access control*). Dabei kann es sich um personenbezogene Daten handeln, die datenschutzrechtlich relevant sind, oder um firmenspezifische Datenbestände. Rechte lassen sich auch auf Sichten definieren.

Transaktionen. Mehrere Datenbankänderungen, die logisch eine Einheit bilden, lassen sich zu Transaktionen zusammenfassen, die als Ganzes ausgeführt werden sollen (Atomarität) und deren Effekt bei Erfolg permanent in der Datenbank gespeichert werden soll (Dauerhaftigkeit).

Mehrbenutzerfähigkeit. Konkurrierende Transaktionen mehrerer Benutzer müssen synchronisiert werden, um gegenseitige Beeinflussung, z. B. bei Schreibkonflikten auf gemeinsam genutzten Daten, zu vermeiden (*concurrency control*). Dem Benutzer erscheinen die Daten so, als ob nur eine Anwendung darauf zugreift (Isolation).

Datensicherung. Das DBMS muss in der Lage sein, bei auftretenden Hard- oder Softwarefehlern wieder einen korrekten Datenbankzustand herzustellen (*Recovery*).

1.3.2 Vorteile des Datenbankeinsatzes

Zusätzlich zu den genannten Merkmalen ergibt sich bei Einsatz eines Datenbanksystems eine Reihe von Vorteilen:

Nutzung von Standards. Der Einsatz einer Datenbank erleichtert Einführung und Umsetzung zentraler Standards in der Datenorganisation. Dies umfasst Namen und Formate von Datenelementen, Schlüssel, Fachbegriffe.

Effizienter Datenzugriff. Ein DBMS gebraucht eine Vielzahl komplizierter Techniken zum effizienten Speichern und Wiederauslesen (retrieval) großer Mengen von Daten.

Kürzere Softwareentwicklungszeiten. Ein DBMS bietet viele wichtige Funktionen, die allen Anwendungen, die auf eine Datenbank zugreifen wollen, gemeinsam ist. In Verbindung mit den angebotenen Datenbanksprachen wird somit eine schnellere Anwendungsentwicklung ermöglicht, da der Programmierer von vielen Routineaufgaben entlastet wird.

Hohe Flexibilität. Die Struktur der Datenbank kann bei sich ändernden Anforderungen ohne große Konsequenzen für die bestehenden Daten und

die vorhandenen Anwendungen modifiziert werden (Datenunabhängigkeit, → 1.4.3).

Hohe Verfügbarkeit. Viele transaktionsintensive Anwendungen (z. B. Reservierungssysteme, Online-Banking) haben hohe Verfügbarkeitsanforderungen. Ein DBMS stellt die Datenbank allen Benutzern dank der Synchronisationseigenschaften *gleichzeitig* zur Verfügung. Änderungen werden nach Transaktionsende sofort sichtbar.

Große Wirtschaftlichkeit. Die durch den Einsatz eines DBMS erzwungene Zentralisierung in einem Unternehmen erlaubt die Investition in leistungsstärkere Hardware, statt jede Abteilung mit einem eigenen (schwächeren) Rechner auszustatten. Somit reduziert der Einsatz eines DBMS die Betriebs- und Verwaltungskosten.

1.3.3 Nachteile von Datenbanksystemen

Trotz der Vorteile eines DBMS gibt es auch Situationen, in denen ein solches System unnötig hohe Zusatzkosten im Vergleich zur traditionellen Dateiverarbeitung mit sich bringen würde:

- Hohe Anfangsinvestitionen für Hardware und Datenbanksoftware.
- Ein DBMS ist Allzweck-Software, somit weniger effizient für spezialisierte Anwendungen.
- Bei konkurrierenden Anforderungen kann das Datenbanksystem nur für einen Teil der Anwendungsprogramme optimiert werden.
- Mehrkosten für die Bereitstellung von Datensicherheit, Mehrbenutzer-Synchronisation und Konsistenzkontrolle.
- Hochqualifiziertes Personal erforderlich, z. B. Datenbankdesigner, Datenbankadministrator.
- Verwundbarkeit durch Zentralisierung (Ausweg Verteilung).

Unter bestimmten Umständen ist der Einsatz regulärer Dateien sinnvoll:

- Ein gleichzeitiger Zugriff auf die Daten durch mehrere Benutzer ist nicht erforderlich.
- Für die Anwendung bestehen feste Echtzeitanforderungen, die von einem DBMS nicht ohne weiteres erfüllt werden können.
- Es handelt sich um Daten und Anwendungen, die einfach und wohldefiniert sind und keinen Änderungen unterliegen werden.

1.3.4 Produkte

Tabelle 1.1 gibt einen Überblick über heute verfügbare DBMS verschiedener Hersteller und die zugrunde liegenden Datenbankmodelle, die eine große Vielfalt aufweisen. Ein „*“ in der Hersteller-Spalte bedeutet, dass das jeweilige DBMS von einem Entwicklerteam verantwortet wird.

Tabelle 1.1 DBMS-Produkte und -Hersteller

DBMS	Hersteller	Modell/Charakteristik
Adabas	Software AG	NF^2-Modell (nicht normalisiert)
Caché	InterSystems	multidimensional, „postrelational“
Cassandra	Apache	NoSQL, spaltenorientiert
CouchDB	Apache	NoSQL, dokumentenorientiert
DB2	IBM	objektrelational
db4o	Versant	objektorientiert, leichtgewichtig
Derby	Apache	relational, Java-basiert, leichtgewichtig
Firebird	*	relational, basierend auf InterBase
IMS	IBM	hierarchisch, Mainframe-DBMS
Informix	IBM	objektrelational
InterBase	Embarcadero	relational
MongoDB	MongoDB	NoSQL, dokumentenorientiert
MS Access	Microsoft	relational, Desktop-System
MS SQL Server	Microsoft	objektrelational
MySQL	Oracle	relational
Neo4J	Neo Technology	NoSQL, Graphdatenbank
Oracle	Oracle	objektrelational
PostgreSQL	*	objektrelational
SAP HANA	SAP	In-Memory-Datenbank
SQLite	*	relational, leichtgewichtig
Sybase ASE	Sybase/SAP	relational
Versant	Versant	objektorientiert
Visual FoxPro	Microsoft	relational, Desktop-System
Teradata	Teradata	relational, massiv parallel

1.4 Architektur eines Datenbanksystems

1.4.1 Architekturen

Der Begriff **Architektur** kann bei Datenbankanwendungen einen unterschiedlichen Kontext haben. Hierbei lassen sich nennen:

- *Systemarchitektur* eines DBMS: Es werden die Komponenten eines DBMS beschrieben, die sich in einem Schichtenmodell anordnen lassen. In Standardisierungsvorschlägen werden die Schnittstellen zwischen den DBMS-Komponenten genormt, nicht jedoch diese selbst.