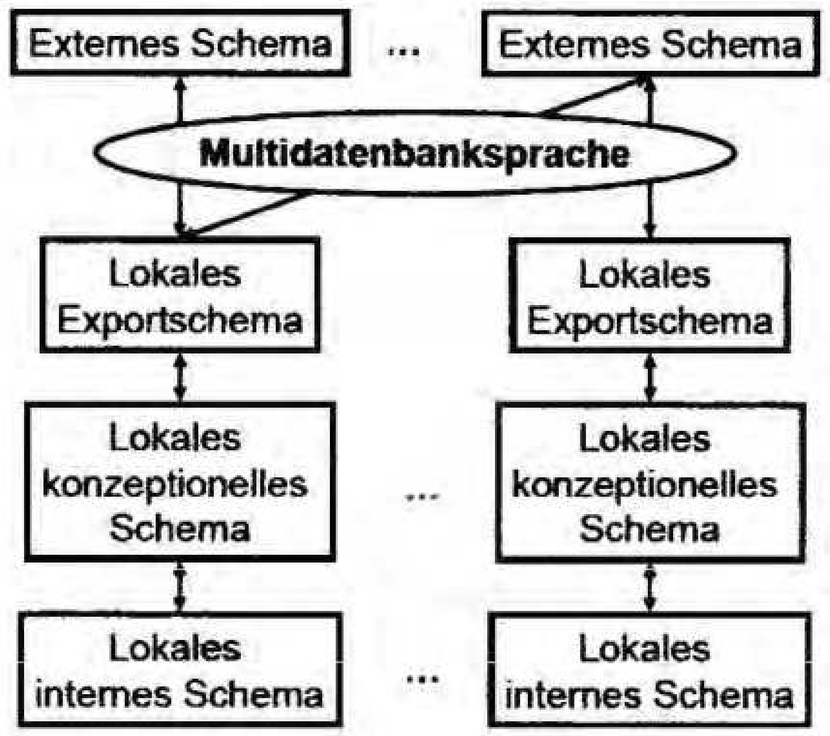

Abbildung 4.4
Die Multidatenbankarchitektur

Anwendung selber überlassen[2]. Das MDBMS stellt lediglich eine Multidatenbanksprache zur Verfügung. Solche Sprachen sind in der Regel an SQL angelehnt und erlauben es, (1) innerhalb einer Anfrage auf mehrere Datenbanken zuzugreifen und (2) strukturelle und semantische Unterschiede in den Schemata zu überbrücken. Wir stellen eine dieser Anfragesprachen, SchemaSQL, in Abschnitt 5.4 vor.

4.4 Föderierte Datenbanksysteme

Fünf-Schichten-Architektur

Im Gegensatz zu MDBMS besitzen föderierte Datenbanksysteme (FDBMS) ein globales konzeptionelles (»föderiertes«) Schema [262]. Dieses Schema ist zentraler und stabiler Bezugspunkt für alle externen Schemata und deren Anwendungen. Im Unterschied zu verteilten Datenbanken entsteht das globale Schema aber nach den lokalen Schemata mit dem Zweck, eine *integrierte Sicht auf existierende und heterogene Datenbestände* zu bieten. In föderierten Datenbanken ist Verteilung der Datenquellen also keinesfalls gewollt, sondern ein unausweichliches Übel. Datenquellen in FDBMS bewahren einen *hohen Grad an Autonomie.*

Ungewollte Verteilung und damit Heterogenität

Kanonisches Datenmodell

Man nennt das im globalen Schema verwendete Datenmodell das *kanonische Datenmodell.* Sämtliche lokalen Exportschemata müssen auf dieses globale Schema abgebildet werden (siehe Abbildung 4.5). Das globale Schema kann auf zwei verschiedene Weisen entstehen: Entweder es wird aus den einzelnen Exportschemata zusammengeführt, oder es wird unabhängig von den lokalen Schemata entworfen. Die erste Variante, die so genannte Schemainte-

[2]Natürlich können entsprechende Anfragen in Sichten zusammengefasst und mehreren Anwendungen zur Verfügung gestellt werden.

gration, wird in Abschnitt 5.1 erläutert. Die zweite Variante, das Schema Mapping, ist Thema von Abschnitt 5.2 und von Kapitel 6.

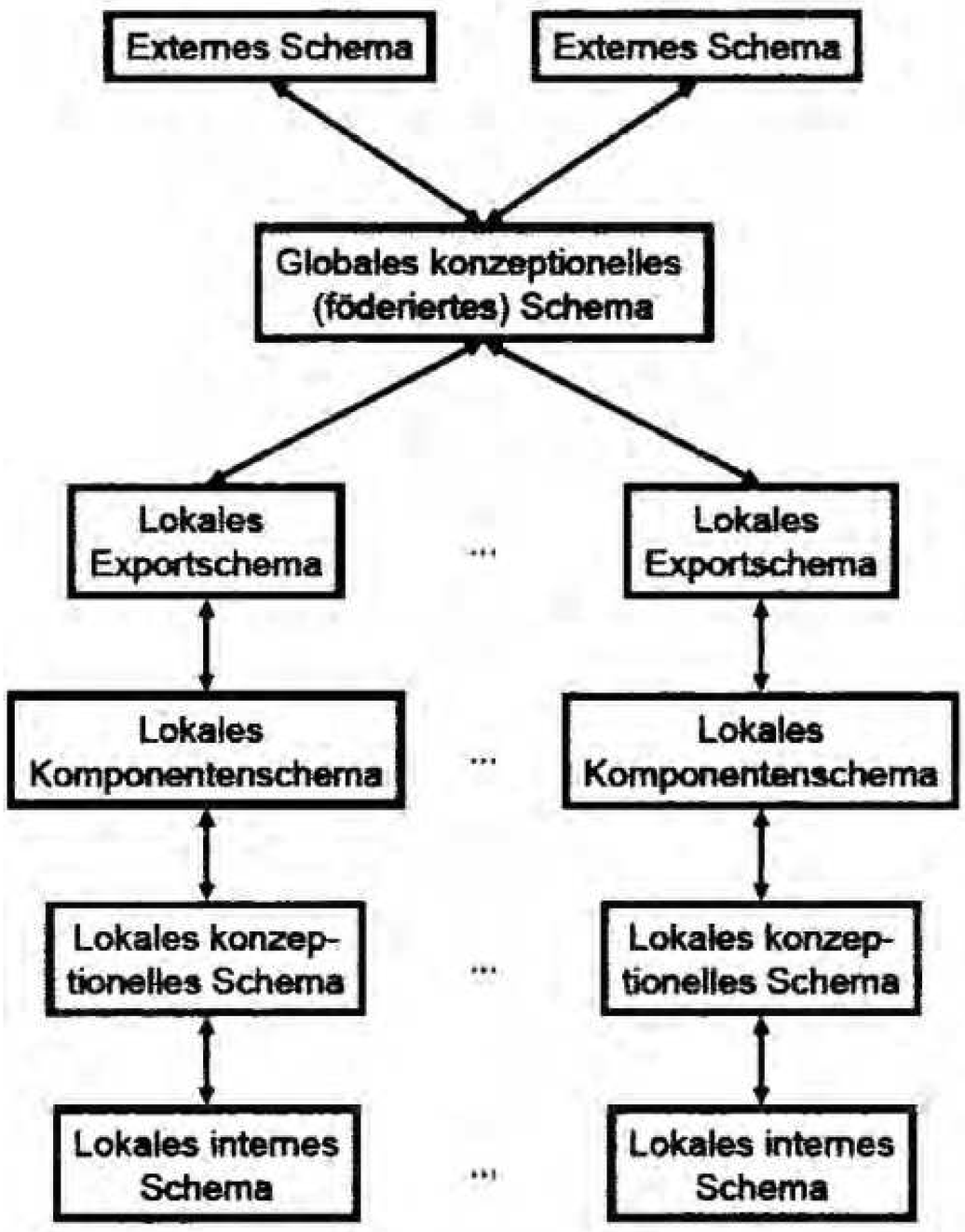

Abbildung 4.5 *Fünf-Schichten-Architektur föderierter Datenbanken (Quelle: [262])*

Da die Fünf-Schichten-Architektur die *Referenzarchitektur* für die meisten in diesem Buch besprochenen Techniken und Systeme ist, erläutern wir hier die Rollen der einzelnen Schichten und Schemata ausführlicher als bisher:

Referenzarchitektur

Schichten in FDBMS

- **Lokales konzeptionelles Schema:** Dieses Schema modelliert die Daten der jeweiligen Datenquelle und spielt die gleiche Rolle wie das konzeptionelle Schema der Drei-Schichten-Architektur. Aufgrund der Autonomie der Quellen kann in Bezug auf das globale Schema sowohl strukturelle Heterogenität als auch Datenmodellheterogenität herrschen.
- **Lokales Komponentenschema:** Diese Schemata modellieren, sofern möglich, die Inhalte des lokalen konzeptionellen Schemas in dem kanonischen Datenmodell, also dem Datenmodell des globalen Schemas. Komponentenschemata überwinden also Datenmodellheterogenität.
- **Lokales Exportschema:** Diese Schemata haben die gleiche Rolle wie in der Drei-Schichten-Architektur: Sie stellen die Teilmenge des Komponentenschemata dar, die nach außen sichtbar ist. Die zugreifende Anwendung ist dabei in erster Linie das FDBMS, obwohl natürlich Anwendungen auch direkt auf die Exportschemata der einzelnen Datenquellen zugreifen können.

- **Globales konzeptionelles Schema:** Dieses Schema hat in der Literatur viele Namen, u.a. globales Schema, föderiertes Schema, Importschema, *enterprise schema* oder integriertes Schema. Es bildet die integrierte Sicht auf alle Datenquellen und ist im kanonischen Datenmodell modelliert. Vorgehensweisen zur Erstellung des globalen Schemas werden in Abschnitt 4.7.1 besprochen.
- **Externes Schema:** Ebenso wie in der Drei-Schichten-Architektur haben die externen Schemata die Aufgabe, diejenigen Teilmengen des globalen Schemas auszuwählen, die für die jeweiligen Anwendungen sichtbar sein sollen.

Im weiteren Verlauf sprechen wir meist nur von lokalen und globalen Schemata. Wir gehen ohne Beschränkung der Allgemeinheit davon aus, dass (1) alle Schemata im gleichen Datenmodell vorliegen und (2) dass das vollständige lokale Schema für das integrierte System sichtbar und zugreifbar ist.

Varianten

In [262] erwähnen die Autoren diverse weitere Architekturvarianten, die im Wesentlichen Kombinationen der hier vorgestellten Varianten sind. Eine wichtige Überlegung in diesem Zusammenhang ist, welches System für welches der Schemata verantwortlich ist und insofern wie autonom die Datenquellen sind. In einer föderierten Datenbank geht man in der Regel davon aus, dass den Datenquellen bekannt ist, dass sie Teil einer Föderation sind und dass sie einen Teil ihrer Autonomie aufgeben. Beispielsweise obliegt es den einzelnen Quellen, ein Komponentenschema zu erstellen, das einzig zum Zweck der Föderation benötigt wird. Die Aufgabe der Erstellung der Exportschemata kann nicht eindeutig zugewiesen werden. Je nach Anwendungsszenario wird die Antwort unterschiedlich ausfallen. In einem rein lesenden Szenario muss die Datenquelle in vielen Fällen gar nicht wissen, dass sie Teil einer Föderation ist. In diesem Fall obliegt es dem integrierten System, das Exportschema zu erstellen. Das automatische Auslesen von Daten aus HTML-Seiten, wie es etwa Metasuchmaschinen betreiben, stellt einen solchen Fall dar. Werden innerhalb einer Organisation verschiedene Datenquellen föderiert, ist es aber eher üblich, dass die Verantwortlichen der Datenquellen selbst das Exportschema definieren. Auf diese Weise kann kontrolliert werden, welcher Teil der Daten global sichtbar wird.