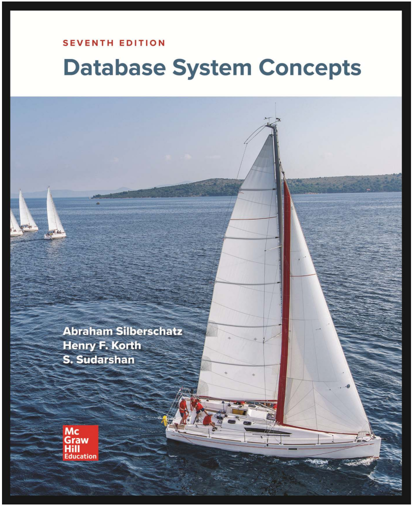
Database System Concepts, 7^{th} Edition
Chapter 1: Introduction

Silberschatz, Korth and Sudarshan

July 22, 2025

Database System Concepts



Content has been extracted from *Database System Concepts*, Seventh Edition, by Silberschatz, Korth and Sudarshan. Mc Graw Hill Education. 2019.
Visit `https://db-book.com/`.

Plan

Database Systems Applications

- DBMS contains information about a particular enterprise:
  - Collection of interrelated data.
  - Set of programs to access the data.
  - An environment that is both *convenient* and *efficient* to use.
- Database systems manage collections of data that are:
  - Highly valuable
  - Relatively large
  - Accessed by multiple users and applications, often at the same time
- A modern database system is a complex software system tasked with managing a large, complex collection of data.
- Databases touch all aspects of our lives.

Database Applications Examples

- Enterprise Information:
  - Sales: customers, products, purchases.
  - Accounting: payments, receipts, assets.
  - Human Resources: Information about employees, salaries, payroll taxes.
  - Manufacturing: management of production, inventory, orders, supply chain.
- Banking and Finance:
  - Customer information, accounts, loans, and banking transactions.
  - Credit card transactions.
  - Finance: sales and purchases of financial instruments (e.g. stocks and bonds, real-time market data, ...)
- Universities:
  - Registration, grades

Database Applications Examples (Cont.)

- Airlines: reservations, schedules
- Telecommunication: records of calls, texts, data usage, generating bills.
- Web-based services:
  - Online retailers: order tracking, customized recommendations.
  - Online advertisements.
- Document databases.
- Navigation systems: locations of places of interest, routes, transport systems, etc.

Plan

Purpose of Database Systems

In the early days, database applications were built directly on top of file systems, which leads to:

- Data redundancy and inconsistency: data is stored in multiple file formats resulting in duplication of information in different files.
- Difficulty in accessing data:
  - New program needed for every new task.
- Data isolation:
  - Multiple files and formats
- Integrity problems:
  - Integrity constraints (e.g., account balance > 0) become "buried" in program code rather than being stated explicitly.
  - Hard to add new constraints or change existing ones.

Purpose of Database Systems (Cont.)

- Atomicity of updates:
  - Failures may leave database in an inconsistent state with partial updates carried out.
  - Example: Transfer of funds from one account to another should either complete or not happen at all.
- Concurrent access by multiple users:
  - Concurrent access needed for performance.
  - Uncontrolled concurrent accesses can lead to inconsistencies.
  - Ex: Two people reading a balance (say 100) and updating it by withdrawing money (say 50 each) at the same time.
- Security problems:
  - Hard to provide user access to some, but not all, data.

Database systems offer solutions to all the above problems.

University Database Example

- We will be using a university database to illustrate all the concepts.
- Data consists of information about:
  - Students
  - Instructors
  - Classes
- Application program examples:
  - Add new students, instructors, and courses.
  - Register students for courses, and generate class rosters.
  - Assign grades to students, compute grade point averages (GPA) and generate transcripts.

Plan

Database Systems Applications

Purpose of Database Systems

View of Data

Data Models

Relational Data Model

Database Languages

Database Design

Database and Application Architecture

Database Users and Administrators

History of Database Systems

View of Data

- A database system is a collection of interrelated data and a set of programs that allow users to access and modify these data.
- A major purpose of a database system is to provide users with an abstract view of the data.
  - Data models:
    - A collection of conceptual tools for describing data, data relationships, data semantics, and consistency constraints.
  - Data abstraction:
    - Hide the complexity of data structures to represent data in the database from users through several levels of data abstraction.

Plan

Data Models

- A collection of tools for describing:
  - Data.
  - Data relationships.
  - Data semantics.
  - Data constraints.
- Relational model.
- Entity-Relationship data model (mainly for database design) .
- Object-based data models (Object-oriented and Object-relational)
- Semi-structured data model (XML).
- Other older models:
  - Network model.
  - Hierarchical model

Plan

Relational Data Model

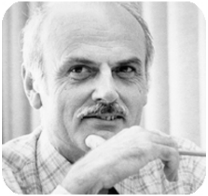
- All the data is stored in various tables.
- Example of tabular data in the relational model:

Column

ID	*name*	*dept_name*	*salary*
22222	Einstein	Physics	95000
12121	Wu	Finance	90000
32343	El Said	History	60000
45565	Katz	Comp. Sci.	75000
98345	Kim	Elec. Eng.	80000
76766	Crick	Biology	72000
10101	Srinivasan	Comp. Sci.	65000
58583	Califieri	History	62000
83821	Brandt	Comp. Sci.	92000
15151	Mozart	Music	40000
33456	Gold	Physics	87000
76543	Singh	Finance	80000

Row

(a) The *instructor* table



Ted Codd
Turing Award 1981

A Sample Relational Database

ID	*name*	*dept_name*	*salary*
22222	Einstein	Physics	95000
12121	Wu	Finance	90000
32343	El Said	History	60000
45565	Katz	Comp. Sci.	75000
98345	Kim	Elec. Eng.	80000
76766	Crick	Biology	72000
10101	Srinivasan	Comp. Sci.	65000
58583	Califieri	History	62000
83821	Brandt	Comp. Sci.	92000
15151	Mozart	Music	40000
33456	Gold	Physics	87000
76543	Singh	Finance	80000

(a) The *instructor* table

dept_name	*building*	*budget*
Comp. Sci.	Taylor	100000
Biology	Watson	90000
Elec. Eng.	Taylor	85000
Music	Packard	80000
Finance	Painter	120000
History	Painter	50000
Physics	Watson	70000

(b) The *department* table

Levels of Abstraction

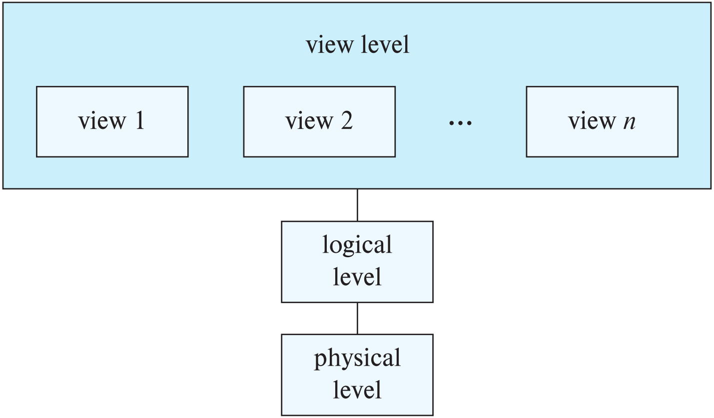
- **Physical level:** describes how a record (e.g., instructor) is stored.
- **Logical level:** describes data stored in database, and the relationships among the data.

```
type instructor = record
    ID: string;
    name: string;
    dept_name: string;
    salary: integer;
end;
```

- **View level:** application programs hide details of data types. Views can also hide information (such as an employee's salary) for security purposes.

Levels of Abstraction

The three levels of data abstraction.



Instances and Schemas

- Similar to types and variables in programming languages.
- **Logical Schema** — the overall logical structure of the database:
  - Example: The database consists of information about a set of customers and accounts in a bank and the relationship between them.
  - Analogous to type information of a variable in a program.
- **Physical Schema** — the overall physical structure of the database.
- **Instance** — the actual content of the database at a particular point in time.
  - Analogous to the value of a variable.

Physical Data Independence

- **Physical Data Independence** — the ability to modify the physical schema without changing the logical schema.
  - Applications depend on the logical schema
  - In general, the interfaces between the various levels and components should be well defined so that changes in some parts do not seriously influence others.

Plan

Data Definition Language (DDL)

- Specification notation for defining the database schema. Example:

```
    CREATE TABLE instructor(
        id varchar(5),
        name varchar(20),
        dept_name varchar(20),
        salary numeric(8,2)
    );
```

Data Definition Language (DDL)

- DDL compiler generates a set of table templates stored in a **data dictionary**.
- Data dictionary contains metadata (i.e., data about data).
  - Database schema.
  - Integrity constraints: Primary key (ID uniquely identifies instructors).
  - Authorization: Who can access what.

Data Manipulation Language (DML)

- Language for accessing and updating the data organized by the appropriate data model.
  - DML also known as query language
- There are basically two types of data-manipulation language:
  - **Procedural DML**
  - **Declarative DML**

Data Manipulation Language (DML)

Aspect	Procedural DML	Declarative DML
Definition	Requires the user to specify *what data* are needed **and how to get them**.	Requires the user to specify *what data* are needed, but **not how to get them**.
Control	Gives more control over the data retrieval process.	Leaves query optimization and data retrieval strategy to the DBMS.
Complexity	More complex to learn and write; suitable for programmers.	Easier to learn and use, especially for end-users and analysts.
Efficiency	Can be more efficient for certain operations if written well, but also more error-prone.	DBMS handles optimization, which may lead to better performance in general.
Example	Navigational access in legacy systems or procedural code embedded in host languages (e.g., C with embedded SQL cursors).	`SELECT name FROM instructor WHERE dept_name = 'History';`

Data Manipulation Language (DML)

- Declarative DMLs are usually easier to learn and use than are procedural DMLs.
- Declarative DMLs are also referred to as non-procedural DMLs.
- The portion of a DML that involves information retrieval is called a query language.

Standard Query Language (SQL)

- SQL query language is nonprocedural. A query takes as input several tables (possibly only one) and always returns a single table.
- Example to find all instructors in Comp. Sci. dept

```sql
SELECT
    name
FROM
    instructor
WHERE
    dept_name = 'Comp. Sci.';
```

Standard Query Language (SQL)

```
SELECT
    instructor.ID, department.dept_name
FROM
    instructor, department
WHERE
    instructor.dept_name = department.dept_name AND
    deptartment.budget >  95000;
```

Standard Query Language (SQL)

- SQL is **NOT** a Turing machine equivalent language.
- To be able to compute complex functions SQL is usually embedded in some higher-level language.
- Application programs generally access databases through one of:
  - Language extensions to allow embedded SQL.
  - Application program interface (e.g., ODBC/JDBC) which allow SQL queries to be sent to a database.

Database Access from Application Program

- Non-procedural query languages such as SQL are not as powerful as a universal Turing machine.
- SQL does not support actions such as input from users, output to displays, or communication over the network.
- Such computations and actions must be written in a host language, such as C/C++, Java or Python, with embedded SQL queries that access the data in the database.
- Application programs — are programs that are used to interact with the database in this fashion.

Plan

Database Design

The process of designing the general structure of the database:

- Logical Design — Deciding on the database schema. Database design requires that we find a “good” collection of relation schemas.
  - Business decision – What attributes should we record in the database?
  - Computer Science decision – What relation schemas should we have and how should the attributes be distributed among the various relation schemas?
- Physical Design — Deciding on the physical layout of the database

Database Engine

- A database system is partitioned into modules that deal with each of the responsibilities of the overall system.
- The functional components of a database system can be divided into:
    - The storage manager.
    - The query processor component.
    - The transaction management component.

Storage Manager

- A program module that provides the interface between the low-level data stored in the database and the application programs and queries submitted to the system.
- The storage manager is responsible to the following tasks:
  - Interaction with the OS file manager.
  - Efficient storing, retrieving and updating of data.
- The storage manager components include:
  - Authorization and integrity manager.
  - Transaction manager.
  - File manager.
  - Buffer manager.

Storage Manager (Cont.)

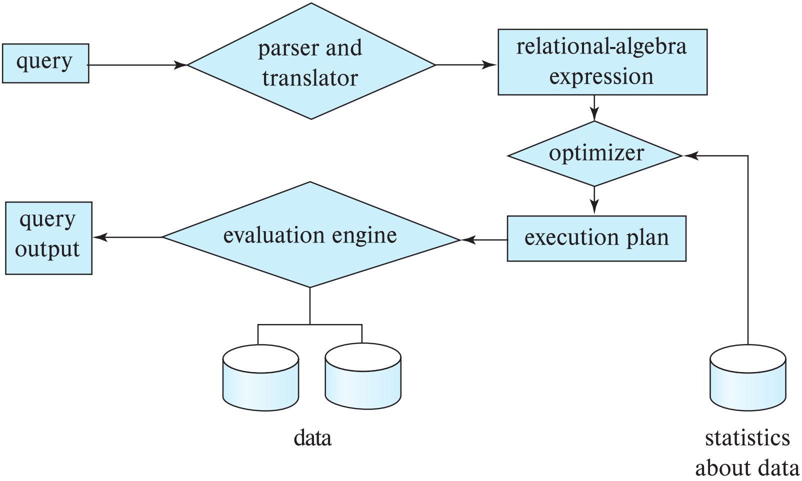
- The storage manager implements several data structures as part of the physical system implementation:
  - Data files – store the database itself.
  - Data dictionary – stores metadata about the structure of the database, in particular the schema of the database.
  - Indices – can provide fast access to data items. A database index provides pointers to those data items that hold a particular value.

Query Processor

- The query processor components include:
  - DDL interpreter – interprets DDL statements and records the definitions in the data dictionary.
  - DML compiler – translates DML statements in a query language into an *evaluation plan* consisting of low-level instructions that the query evaluation engine understands.
    - The DML compiler performs query optimization; that is, it picks the lowest cost evaluation plan from among the various alternatives.
  - Query evaluation engine – executes low-level instructions generated by the DML compiler.

Query Processing

- Parsing and translation.
- Optimization.
- Evaluation.



Transaction Management

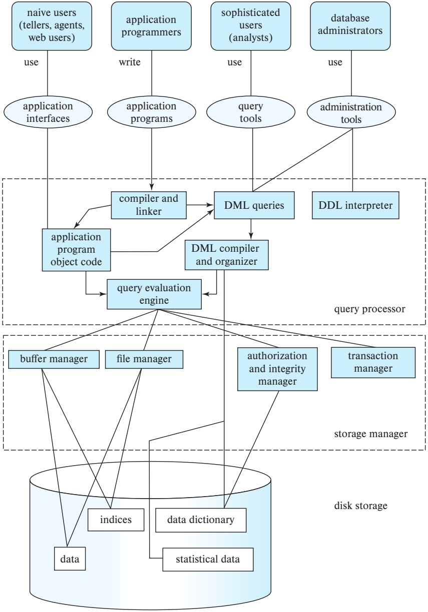
- A **transaction** is a collection of operations that performs a single logical function in a database application
- **Transaction-management component** ensures that the database remains in a consistent (correct) state despite system failures (e.g., power failures and operating system crashes) and transaction failures.
- **Concurrency-control manager** controls the interaction among the concurrent transactions, to ensure the consistency of the database.

Plan

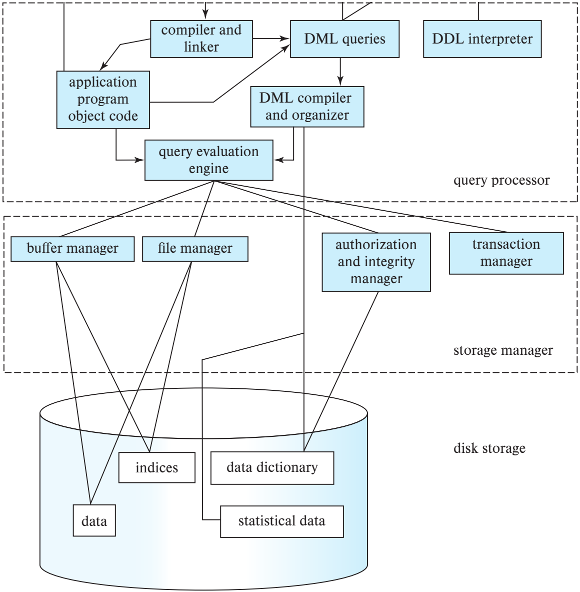
Database Architecture

- Centralized databases:
  - One to a few cores, shared memory.
- Client-server:
  - One server machine executes work on behalf of multiple client machines.
- Parallel databases:
  - Many core shared memory.
  - Shared disk.
  - Shared nothing.
- Distributed databases:
  - Geographical distribution.
  - Schema/data heterogeneity.

Database Architecture (Centralized/Shared-Memory)



Database Architecture (Centralized/Shared-Memory)

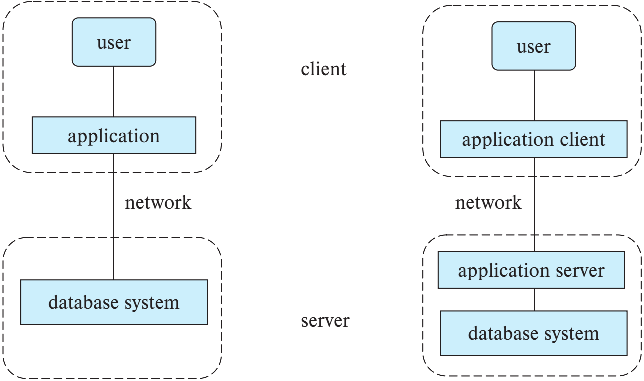


Database Applications

Database applications are usually partitioned into two or three parts:

- Two-tier architecture – the application resides at the client machine, where it invokes database system functionality at the server machine.
- Three-tier architecture – the client machine acts as a front end and does not contain any direct database calls.
  - The client end communicates with an application server, usually through a forms interface.
  - The application server in turn communicates with a database system to access data.

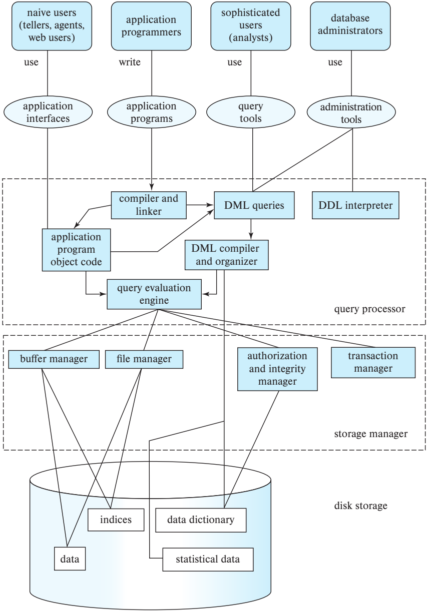
Two-tier and three-tier architectures



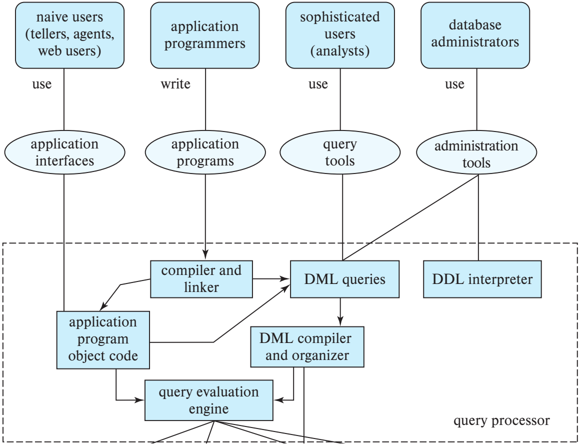
(a) Two-tier architecture (b) Three-tier architecture

Plan

Database Users and User Interfaces



Database Users and User Interfaces



Database Administrator

A person who has central control over the system is called a **Database Administrator (DBA)**. Functions of a DBA include:

- Schema definition.
- Storage structure and access-method definition.
- Schema and physical-organization modification.
- Granting of authorization for data access.
- Routine maintenance.
- Periodically backing up the database.
- Ensuring that enough free disk space is available for normal operations, and upgrading disk space as required.
- Monitoring jobs running on the database.

Plan

- Database Systems Applications
- Purpose of Database Systems
- View of Data
- Data Models
- Relational Data Model
- Database Languages
- Database Design
- Database and Application Architecture
- Database Users and Administrators
- History of Database Systems

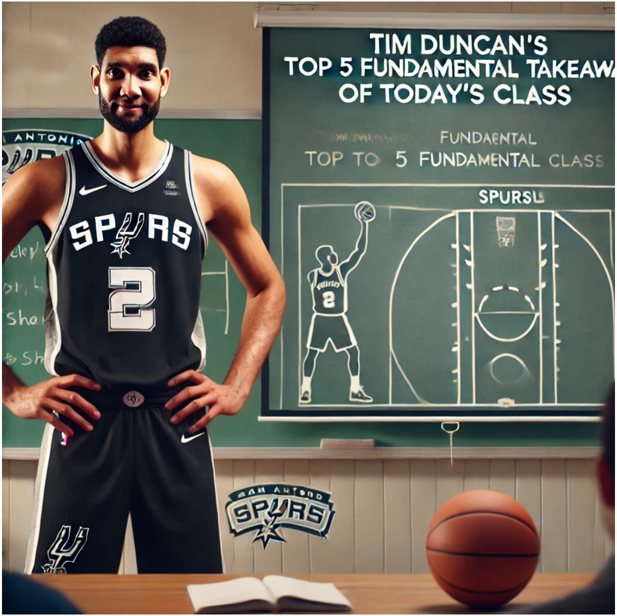
History of Database Systems

- 1950s and 60s: Magnetic tapes and punched cards...
- 1970s: Relational model introduced by Ted Codd...
- 1980s: Commercialization of relational databases...
- 1990s: Data warehouses and data mining...
- 2000s: Big data systems (NoSQL, MapReduce)...

TDT5FTOTC

Tim Duncan in Wikipedia.

TDT5FTOTTC


TIM DUNCAN'S
TOP 5 FUNDAMENTAL TAKEAW
OF TODAY'S CLASS
FUNDAENTAL
TOP TO 5 FUNDAMENTAL CLASS
SPURS
SPURS
2
SPURS

Top 5 Fundamental Takeaways

Top 5 Fundamental Takeaways

5. **Evolution of DBMS:** Database systems have advanced from simple file-processing systems to distributed and cloud-based solutions.

Top 5 Fundamental Takeaways

5 **Evolution of DBMS:** Database systems have advanced from simple file-processing systems to distributed and cloud-based solutions.

4 **Purpose of DBMS:** A DBMS ensures efficient, secure, and reliable management of data while addressing redundancy and inconsistency.

Top 5 Fundamental Takeaways

5 **Evolution of DBMS:** Database systems have advanced from simple file-processing systems to distributed and cloud-based solutions.

4 **Purpose of DBMS:** A DBMS ensures efficient, secure, and reliable management of data while addressing redundancy and inconsistency.

3 **Database Applications:** Databases are integral to transaction processing, data analytics, and modern big data environments.

Top 5 Fundamental Takeaways

5 **Evolution of DBMS:** Database systems have advanced from simple file-processing systems to distributed and cloud-based solutions.

4 **Purpose of DBMS:** A DBMS ensures efficient, secure, and reliable management of data while addressing redundancy and inconsistency.

3 **Database Applications:** Databases are integral to transaction processing, data analytics, and modern big data environments.

2 **Database Languages:** DBMS uses DDL for defining structures and DML for querying and modifying data effectively.

Top 5 Fundamental Takeaways

5 **Evolution of DBMS:** Database systems have advanced from simple file-processing systems to distributed and cloud-based solutions.

4 **Purpose of DBMS:** A DBMS ensures efficient, secure, and reliable management of data while addressing redundancy and inconsistency.

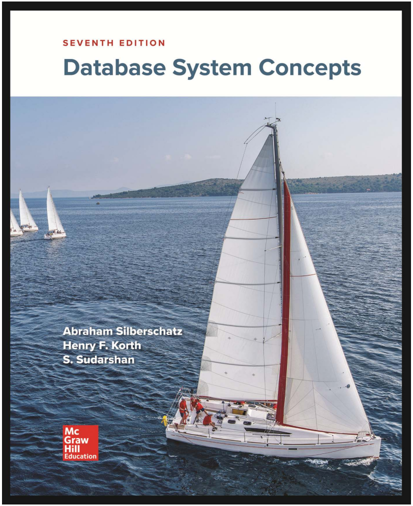
3 **Database Applications:** Databases are integral to transaction processing, data analytics, and modern big data environments.

2 **Database Languages:** DBMS uses DDL for defining structures and DML for querying and modifying data effectively.

1 **Data Abstraction:** It provides physical, logical, and view-level abstractions to simplify user interaction and system complexity.

End of Chapter 1.

Database System Concepts



Content has been extracted from *Database System Concepts*, Seventh Edition, by Silberschatz, Korth and Sudarshan. Mc Graw Hill Education. 2019.
Visit `https://db-book.com/`.