2.1 Database basics

Data

Data is numeric, textual, visual, or audio information that describes real-world systems. Data is collected and processed to aid in a variety of tasks, such as forecasting weather, analyzing financial investments, and tracking the global spread of pandemics.

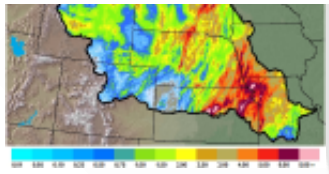
Data can vary in several important ways:

- *Scope.* The amount of data produced and collected can vary. Ex: A small business might track an inventory of a few thousand items, but a large commerce website might track billions of items.
- *Format.* Data may be produced as numbers, text, image, audio, or video. Ex: A phone's proximity sensor generates raw numbers, and a satellite captures images.
- *Access.* Some data sources are private while others are made publicly available. Ex: A retail company may use private customer data to discover purchasing behavior patterns, but a government may be required by law to share certain data sets.

Historically, data was mostly ***analog***, encoded as continuous variations on various physical media. Ex: Audio was recorded as vibrations impressed on vinyl disks. Images were recorded as chemicals on celluloid tapes. Today, data is mostly ***digital***, encoded as zeros and ones on electronic and magnetic media.

The shift from analog to digital data facilitated the rise of large computer databases.

PARTICIPATION ACTIVITY 2.1.1: Examples of public data sets.



data.Nasa.gov

Light Measurements

9.23,9.23,9.77,9.22,8.98
10.20,7.85,8.83,9.44
6.12,8.85,9.03,8.50,1.33

Animation content:

Static figure:
Three public data sets appear. Each data set has a source web site, a caption describing the data set, and several rows of example data.

The first data set is from the website data.gov. The data set caption is US Hourly Precipitation Data. The example data is:
Denver: 4.5, 3.4, 7.4, 3.5
Austin: 0.3, 2.1, 2.4, 0.6
Little Rock: 5.2, 6.1, 0.2, 3.5

The second data set is from the website kaggle.com. The caption is Financial Tweets. The example data is:
@StockTwits The price of lumber was ...
@ThStreet Macys $2M turnaround ...
@MarketWatch Silbert on #bitcoin is ...
@JimCramer Liked @RealMoney last ...

The third data set is from the website data.nasa.gov. The caption is Light Measurements. The example data is:
9.23, 9.23, 9.77, 9.22, 8.98
10.20, 7.85, 8.83, 9.44
6.12, 8.85, 9.03, 8.50, 1.33

Animation captions:

1. Data.gov provides thousands of U.S. government data sets. Precipitation data can be used to visualize rainfall intensity.
2. Kaggle.com allows users to find and publish data sets. The Financial Tweet data set can show who tweets on similar topics.
3. data.Nasa.gov provides data sets in aerospace and other related sciences. A data set of light measurements describes astronomical phenomena.

Sources: Rainfall map, Social network diagram, Hubble mosaic of the Crab Nebula from Wikipedia.org

PARTICIPATION ACTIVITY 2.1.2: Public data sets.

These websites offer public data sets:

- data.gov
- cancer.gov/research
- kaggle.com
- data.nasa.gov
- opendata.cityofnewyork.us

Click on each link, review the website, and drag the website name below to the matching description.

If unable to drag and drop, refresh the page.

kaggle.com **data.nasa.gov** **opendata.cityofnewyork.us** **cancer.gov/research** **data.gov**

	Provides more than 250,000 U.S. government data sets to support research and application development.
	Collects and reports data and information relative to all forms of cancer.
	Owned by Google, supports an

	online community that allows users to find and publish data sets.
	Provides numerous data sets in categories such as aerospace, earth science, and space science.
	Data collected by the New York City government to support continuous monitoring and improvements to NYC and residents' health.

Reset

Databases

A ***database*** is a collection of data in a structured format. In principle, databases can be stored on paper or even clay tablets. In practice, however, modern databases are invariably stored on computers. The database structure ensures that similar data is stored in a standardized manner.

Many modern databases contain trillions of bytes of data and support thousands of simultaneous users. Consequently, databases must be managed with sophisticated software tools:

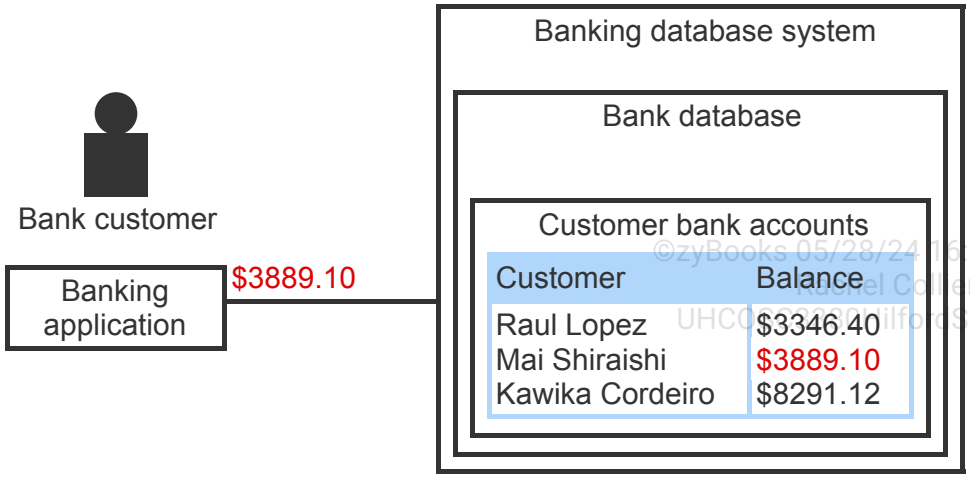
- A ***database system***, also known as a ***database management system*** or ***DBMS***, is software that reads and writes data in a database. Database systems ensure data is secure, internally consistent, and available at all times. These functions are challenging for large databases with many users, so database systems are complex.
- A ***query*** is a request to retrieve or change data in a database. A ***query language*** is a specialized programming language, designed specifically for database systems. Query languages read and write data efficiently, and differ significantly from general-purpose languages such as Python, Java, and C++.
- A ***database application*** is software that helps business users interact with database systems. Many databases are complex, and most users are not familiar with query languages. Consequently, direct database access is usually not feasible. Instead, programmers write applications to simplify the user experience and ensure data access is efficient and secure.

Database software is organized in layers. Applications interact with a query language on one layer, and a query language interacts with a database system on another layer. Other software layers, such as the operating system, are beyond the scope of this material.

Terminology

*The term **database** sometimes refers to a database system rather than the data managed by the system. The meaning is usually clear from context. In this material, **database** is used both ways.*

PARTICIPATION ACTIVITY 2.1.3: A bank database.



Animation content:

Static figure:
Three rectangles appear with captions banking database system, bank database, and customer bank account. The banking database system contains the banking database, which contains customer bank account. A table of customer names and account balances appears inside customer bank account.
A fourth rectangle has caption banking application, and is connected to the banking database system with a line.
The animation illustrates a deposit moving from the banking application to the banking database system, followed by an increase in the account balance for one customer name. The banking database system then reports the new account balance to the banking application.

Animation captions:

1. Banking data is stored in a database and is managed by a database system.
2. A bank customer uses an application to perform bank transactions.

3. Banking transactions cause the database system to modify the bank's database and update the user's account.

PARTICIPATION ACTIVITY 2.1.4: Databases.

Refer to the animation above.

1) Where would a bank customer's account data be directly stored?

- In the Banking database system
- In the customer banking application
- In the Bank database

2) Which of the following manages the Bank database?

- The customer banking application
- The banking database system
- The computer the Banking database system resides on

3) Which of the following would prevent unauthorized access to the Bank database?

- The banking database system
- The Bank database
- The customer banking application

4) How would a bank customer access their bank data?

- Through the banking database system
- Through the customer banking application

- Through the Bank Database

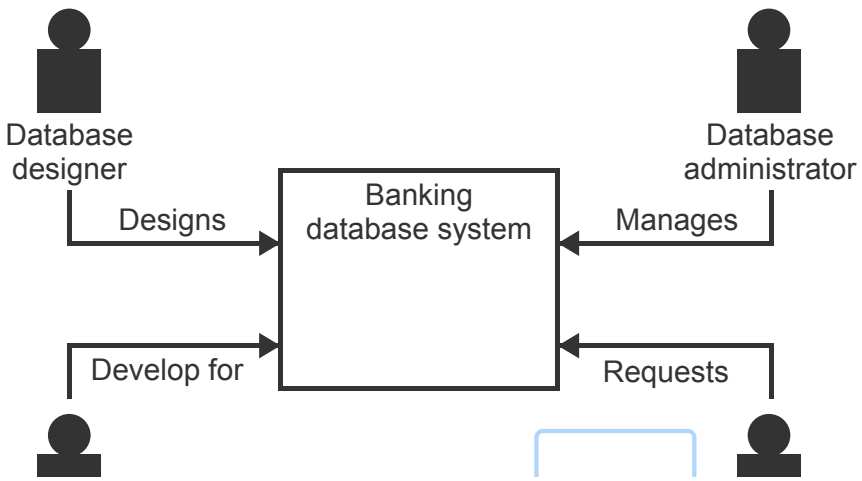
Database roles

People interact with databases in a variety of roles:

- A ***database administrator*** is responsible for securing the database system against unauthorized users. A database administrator enforces procedures for user access and database system availability.
- A ***database designer*** determines the format of each data element and the overall database structure. Database designers must balance several priorities, including storage, response time, and support for rules that govern the data. Since these priorities often conflict, database design is technically challenging.
- A ***database programmer*** develops computer programs that utilize a database. Database programmers write applications that combine database query languages and general-purpose programming languages. Query languages and general-purpose languages have significant differences, so database programming is a specialized challenge.
- A ***database user*** is a consumer of data in a database. Database users request, update, or use stored data to generate reports or information. Database users usually access the database via applications but can also submit queries directly to the database system.

For simple databases with a limited amount of data and few users, one person may assume several roles. Ex: The database administrator might also be a database designer or database programmer. For large, complex databases, each person usually takes on just one role.

PARTICIPATION ACTIVITY 2.1.5: Database roles.



Animation content:

Static figure:
A box labeled banking database system appears. The box is surrounded by four icons, each representing a person and connected to the banking database system with an arrow. The four icon labels are database administrator, database designer, database programmer, and database user.

Step 1: The database designer establishes the structure of the database and determines the data to be collected and stored. The words create bank database move from the database designer to the banking database system. The words create customer and account move from the database designer to the banking database system.

Step 2: The database administrator ensures the database is available and secure. The words add user move from the database administrator to the banking database system. The words backup data move from the database administrator to the banking database system.

Step 3: A database programmer uses query languages and programming languages to develop applications for database users. The words query database move from the database programmer to the banking database system. The words alter data move from the database programmer to the banking database system.

Step 4: Database users are the primary consumers of database data through applications and query languages. The words change address move from the database user to the banking database system. The words transfer money move from the database user to the banking database system.

Animation captions:

1. The database designer establishes the structure of the database and determines the data to be collected and stored.
2. The database administrator ensures the database is available and secure.
3. A database programmer uses query languages and programming languages to develop applications for database users.
4. Database users are the primary consumers of database data through applications and query languages.

PARTICIPATION ACTIVITY 2.1.6: Roles.

1) Which role is responsible for providing access to the database?
   - ○ Database administrator
   - ○ Database designer
   - ○ Database programmer
   - ○ Database users

2) Which role is responsible for defining the detailed database design?
   - ○ Database administrator
   - ○ Database designer
   - ○ Database programmer
   - ○ Database users

3) Which role uses an application to query a database and generate a report?
   - ○ Database designer
   - ○ Database administrator
   - ○ Database programmer
   - ○ Database users

Exploring further:

- Data and Reality, by William Kent
- Data.gov
- Kaggle
- Data.NASA.gov

2.2 Database systems

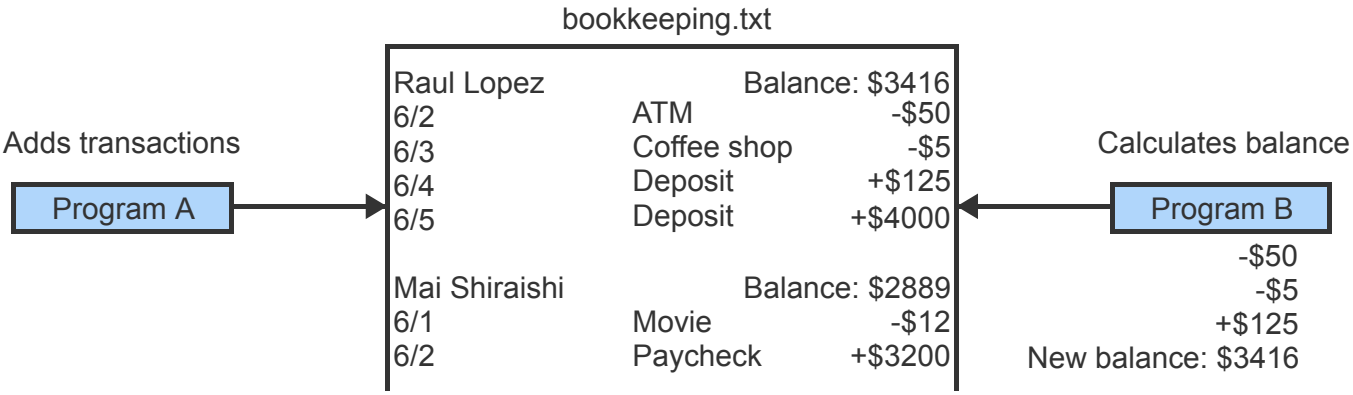
File systems and database systems

Small databases that are shared by one or two users can be managed in a text file or spreadsheet. Text files and spreadsheets are inadequate, however, as databases grow in size, complexity, and use. Large, complex databases that are shared by many users have special requirements:

- *Performance*. When many users and applications simultaneously access large databases, query response time degrades rapidly. Database systems maintain fast response times by structuring data properly on storage media and processing queries efficiently.
- *Authorization*. Many database users should have limited access to specific tables, columns, or rows of a database. Database systems authorize individual users to access specific data.
- *Security*. Database systems ensure authorized users only access permissible data. Database systems also protect against hackers by encrypting data and restricting access.
- *Rules*. Database systems ensure data is consistent with structural and business rules. Ex: When multiple copies of data are stored in different locations, copies must be synchronized as data is updated. Ex: When a course number appears in a student registration record, the course must exist in the course catalog.
- *Recovery*. Computers, database systems, and individual transactions occasionally fail. Database systems must recover from failures and restore the database to a consistent state without loss of data.

File systems are not designed for these demanding requirements. The limitations of file systems became clear as business adopted computers in the 1960s. Since then, database systems have replaced file systems for large databases with many users.

PARTICIPATION ACTIVITY 2.2.1: Limitations of file systems.



Kawika Cordeiro		Balance: $8291
6/1	Groceries	-$58
6/3	Phone	-$45
13/31	mbflwx	+6z@yy
6/4	Deposit	+$999999

Animation content:

Static figure:
A file named bookkeeping.txt contains banking transaction data for three customers. For each customer, the file shows account balance and several banking transactions. Each banking transaction has a date, description, and amount.

A rectangle is named Program A with caption adds transactions. Another rectangle is named Program B with caption calculates balance. Arrows point from both rectangles to bookkeeping.txt. Off to the side, an icon of a person has caption unauthorized user.

Step 1: A list of bank transactions is stored in the text file bookkeeping.txt. Each transaction includes a date, type, and the amount of money paid or received.

Step 2: Two programs access the text file. One adds new transactions, and the other calculates account balances.

Step 3: When program A writes transactions quickly to the file, Program B misses the $4000 deposit and calculates Raul's balance incorrectly. Several transactions for one customer, Raul Lopez, move quickly from Program A to bookkeeping.txt. Program B calculates the new balance for Raul Lopez, but misses the last transaction and calculates the wrong balance, which is stored in bookkeeping.txt.

Step 4: Program A may write an erroneous transaction. A transaction for another customer, Kawika Cordeiro, moves from Program A to bookkeeping.text. The transaction contains an invalid (non-numeric) amount but, nevertheless, is recorded for Kawika Cordeiro in bookkeeping.txt.

Step 5: A lack of adequate security could allow unauthorized users to access the file. A transaction moves from the unauthorized user and is recorded in bookkeeping.txt for Kawida Cordeiro.

Animation captions:

1. A list of bank transactions is stored in the text file bookkeeping.txt. Each persons' transactions include a date, type, and the amount of money paid or received.
2. Two programs access the text file. One adds new transactions, and the other calculates

2. Two programs access the text file. One adds new transactions, and the other calculates account balances.
3. When program A writes transactions quickly to the file, Program B misses the $4000 deposit and calculates Raul's balance incorrectly.
4. Program A may write an erroneous transaction.
5. A lack of adequate security could allow unauthorized users to access the file.

PARTICIPATION ACTIVITY 2.2.2: Limitations of file systems.

Refer to the animation above.

1) If program A writes to bookkeeping.txt the same time that program B reads from bookkeeping.txt, what can potentially go wrong?

- Nothing can go wrong.
- Program A may be writing only partial data.
- Program B may be reading only partial data.

2) Why was an unauthorized user permitted to perform the transaction that added $999999 to bookkeeping.txt?

- The user had read access to the text file.
- The user had write access to the text file.
- The user had no access to the text file.

3) What is wrong with the transaction posted on 13/31 for the amount +6z@yy?

- The data in the transaction contains invalid pieces of data.

- The transaction data is valid and would correctly update the bookkeeping.txt file.

Transactions

Transaction management is a particularly challenging requirement for database systems.

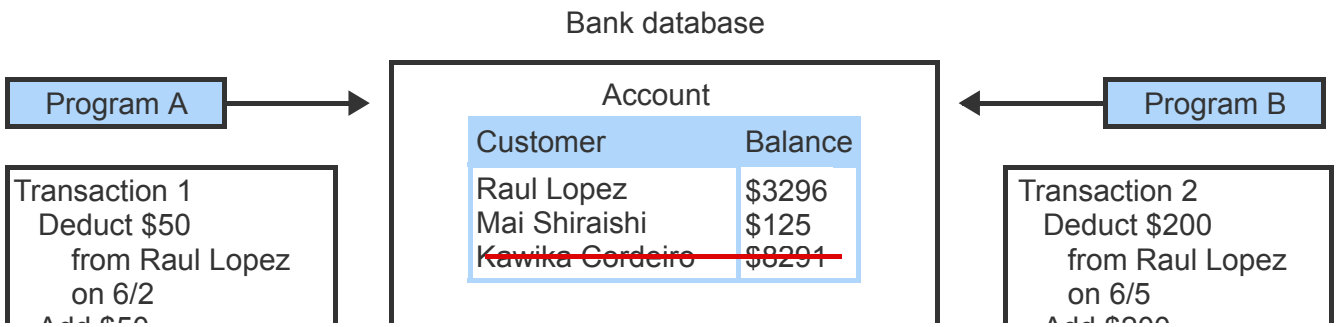
A ***transaction*** is a group of queries that must be either completed or rejected as a whole. Execution of some, but not all, queries results in inconsistent or incorrect data. Ex: A debit-credit transaction transfers funds from one bank account to another. The first query removes $100 from one account and the second query deposits $100 in another account. If the first query succeeds but the second fails, $100 is mysteriously lost. The transaction must process either both queries or neither query.

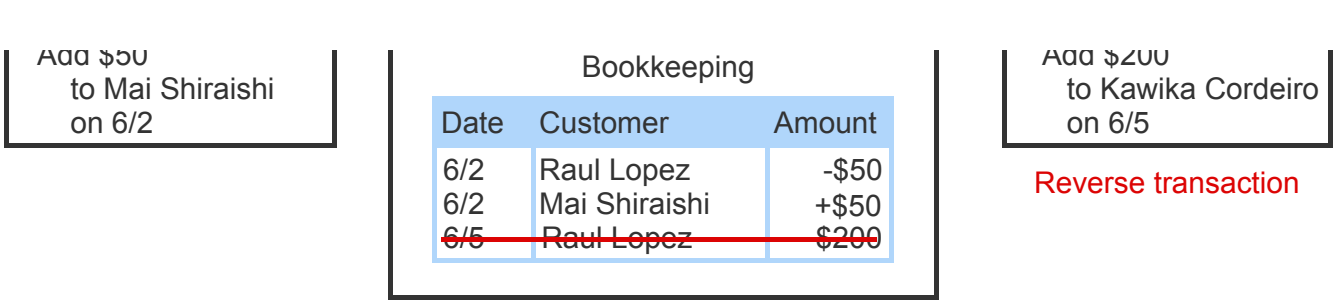
When processing transactions, database systems must:

- *Ensure transactions are processed completely or not at all*. A computer or application might fail while processing a transaction. When failing to process a transaction, the database system must reverse partial results and restore the database to the values prior to the transaction.
- *Prevent conflicts between concurrent transactions*. When multiple transactions access the same data at the same time, a conflict may occur. Ex: Sam selects a seat on a flight. Maria purchases the same seat in a separate transaction before Sam completes his transaction. When Sam clicks the 'purchase' button, his seat is suddenly unavailable.
- *Ensure transaction results are never lost*. Once a transaction completes, transaction results must always be saved on storage media, regardless of application or computer failures.

The above requirements are supported in sophisticated transaction management subsystems of most database systems.

PARTICIPATION ACTIVITY 2.2.3: Transactions.





Animation content:

Static figure:
A bank database contains two tables. The first table is named Account and has columns Customer and Balance. The second table is named Bookkeeping and has columns Date, Customer, and Amount. Both tables have one row of data for Raul Lopez, one for Mai Shiraishi, and one for Kawika Cordeiro.

Boxes named program A and program B appear. Both boxes have an arrow to the bank database.

Below program A is a box named transaction 1 with two entries. The first entry is deduct $50 from Raul Lopez on 6/2. The second entry is add $50 to Mai Shiraishi on 6/2.

Below program B is a box labeled transaction 2 with two entries. The first entry is deduct $200 from Raul Lopez on 6/5. The second entry is add $200 to Kawika Cordeiro on 6/5. Below transaction 2 is the caption reverse transaction.

Step 1: Two programs access a bank database. The database tracks customer deposits, credits, and account balances. The account table shows entries for each of the three customers. The Bookkeeping table is empty. The transactions do not yet appear under program A and program B.

Step 2: Program A requests the database transfer $50 from Raul to Mai. Transaction 1 appears under program A.

Step 3: Transaction 1 deducts $50 from Raul's account and adds $50 to Mai's account. The two entries of transaction 1 appear as rows in the Bookkeeping table. The balances in the Account table change to reflect the new rows of the Bookkeeping table.

Step 4: Program B requests that Raul transfer $200 to Kawika. Transaction 2 appears under program B.

Step 5: $200 is deducted from Raul's account. A third row is added to the Bookkeeping table, showing the deduction of $200 from the Raul Lopez account. The balance for Raul Lopez is decreased by $200 in the Account table.

Step 6: Kawika closes his account before Transaction 2 completes. The database reverses the $200 deduction. The row for Kawika Cordeiro is crossed out in the Account table. The caption reverse transaction appears under transaction 2. The row of the Bookkeeping table, showing a deduction of $200 from Raul Lopez account, is crossed out. The balance for Raul Lopez in the Account table is increased by $200.

Animation captions:

1. Two programs access a bank database. The database tracks customer deposits, credits, and account balances.
2. Program A requests the database transfer $50 from Raul to Mai.
3. Transaction 1 deducts $50 from Raul's account and adds $50 to Mai's account.
4. Program B requests that Raul transfer $200 to Kawika.
5. $200 is deducted from Raul's account.
6. Kawika closes his account before Transaction 2 completes. The database reverses the $200 deduction.

PARTICIPATION ACTIVITY 2.2.4: Transactions.

Match the transaction behavior to the described situation.

If unable to drag and drop, refresh the page.

Prevent conflicts between concurrent transactions

Transaction processed completely or not at all

Ensure transaction results are never lost

	A program is adding a penalty fee to an account that is below $1000 while another program is adding $2000 to the same account.
	A streaming video service's database is disabled due to a lightning strike. The IT department

	verifies that all database transactions performed before the lightning strike were saved on storage media.
	Maria purchases an airline ticket, but a server failure causes the ticket to become unavailable before the transaction completes. The database must reverse any partial changes.

Reset

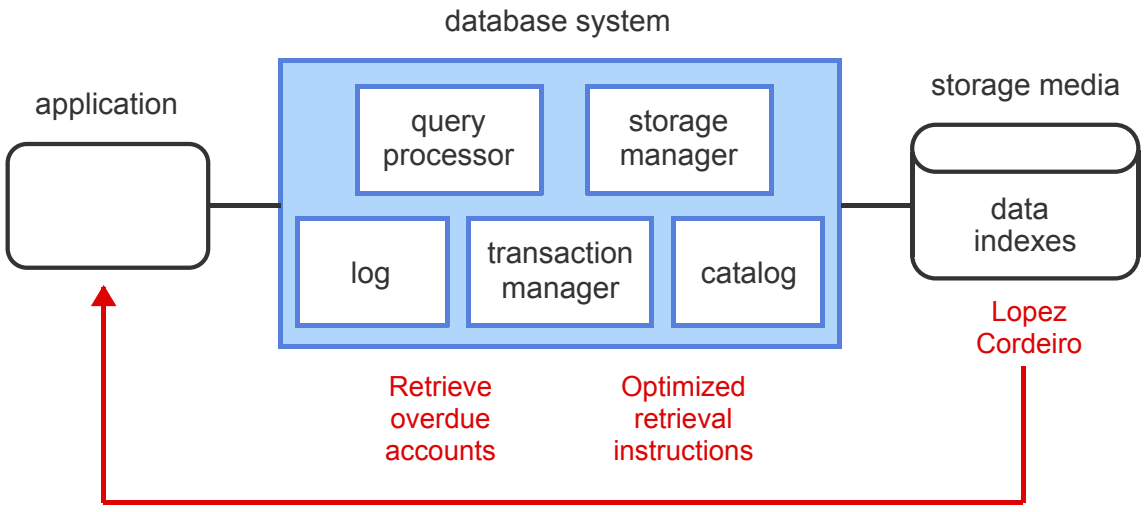
Architecture

The ***architecture*** of a database system describes the internal components and the relationships between components. At a high level, the components of most database systems are similar:

- The ***query processor*** interprets queries, creates a plan to modify the database or retrieve data, and returns query results to the application. The query processor performs ***query optimization*** to ensure the most efficient instructions are executed on the data.
- The ***storage manager*** translates the query processor instructions into low-level file-system commands that modify or retrieve data. Database sizes range from megabytes to many terabytes, so the storage manager uses ***indexes*** to quickly locate data.
- The ***transaction manager*** ensures transactions are properly executed. The transaction manager prevents conflicts between concurrent transactions. The transaction manager also restores the database to a consistent state in the event of a transaction or system failure.
- The ***log*** is a file containing a complete record of all inserts, updates, and deletes processed by the database. The transaction manager writes log records before applying changes to the database. In the event of a failure, the transaction manager uses log records to restore the database.
- The ***catalog***, also known as a ***data dictionary***, is a directory of tables, columns, indexes, and other database objects. Other components use catalog information to process and execute queries.

Database systems have different capabilities, and component details vary significantly. Ex: Some database systems do not support transactions and therefore have no transaction manager. Ex: The storage manager implementation depends on the physical structure of data on storage media.

PARTICIPATION ACTIVITY 2.2.5: Database system architecture.



Animation content:

Static figure:
A box named database system contains boxes named transaction manager, query processor, storage manager, log, and catalog. The database system is connected by lines to a box named application and to a box named storage media.

Step 1: A database system is composed of a query processor, storage manager, transaction manager, log, and catalog. The database system and internal boxes appear.

Step 2: An application sends queries to the query processor. The application box appears. Text appears next to the application that states retrieve overdue accounts. The text moves from the application to below the database system. The query processor is highlighted.

Step 3: The query processor uses information from the catalog to perform query optimization. New text appears below the database system that states optimized retrieval instructions. The query processor and catalog are highlighted.

Step 4: The storage manager translates the query processor instructions into file system commands and uses an index to quickly locate the requested data. The storage media box appears, containing the words data indexes. The word query moves from the database system to storage media. Lopez Cordeiro appears below storage media. The storage manager and catalog are highlighted.

Step 5: The transaction manager logs insert, update, and delete queries, and the result is sent back to the application. An arrow appears that goes from storage media to the application. The words Lopez Cordiero move along the line to the application. The transaction manager and log are highlighted.

Animation captions:

1. A database system is composed of a query processor, storage manager, transaction manager, log, and catalog.
2. An application sends queries to the query processor.
3. The query processor uses information from the catalog to perform query optimization.
4. The storage manager translates the query processor instructions into file system commands and uses an index to quickly locate the requested data.
5. The transaction manager logs insert, update, and delete queries, and the result is sent back to the application.

PARTICIPATION ACTIVITY 2.2.6: Database system architecture.

1) The query processor has direct access to the database data on storage media.
   - True
   - False

2) Without query optimization, the storage manager cannot retrieve the database data.
   - True
   - False

3) The catalog allows the storage manager to quickly locate the requested data.
   - True
   - False

4) Every database query must be logged by the transaction manager to recover

the database in the event of a system failure.

- ○ True
- ○ False

Products

Most leading database systems are relational. A ***relational database*** stores data in tables, columns, and rows, similar to a spreadsheet. All data in a column has the same format. All data in a row represents a single object, such as a person, place, product, or activity.

All relational database systems support the SQL query language. ***SQL*** stands for Structured Query Language and includes statements that read and write data, create and delete tables, and administer the database system.

Relational systems are ideal for databases that require an accurate record of every transaction, such as banking, airline reservation systems, and student records. The growth of the internet in the 1990s generated massive volumes of online data, called ***big data***, often with poorly structured or missing information. Relational systems were not initially designed for big data and, as a result, many non-relational systems have appeared since 2000. The newer non-relational systems are called ***NoSQL***, for 'not only SQL', and are optimized for big data.

Prior to 2000, most database systems were commercial products, developed by for-profit companies and licensed for a fee. Since 2000, an alternative licensing model, called open source, has become popular. ***Open source*** software is software that anyone can inspect, copy, and modify with no licensing fee.

NoSQL and open source systems have proliferated, and hundreds of database systems are now available. The website db-engines.com ranks systems by tracking product references on social media, internet searches, job websites, and technical websites. Internet references are an imperfect measure of product utilization, but do provide a general indication of interest and activity.

Figure 2.2.1: Leading database products.

Product	Sponsor	Type	License	DB-Engines rank (May 2020)
Oracle Database	Oracle	Relational	Commercial	1

MySQL	Oracle	Relational	Open source	2
SQL Server	Microsoft	Relational	Commercial	3
PostgreSQL	PostgreSQL Global Development Group	Relational	Open source	4
MongoDB	MongoDB	NoSQL	Open source	5

PARTICIPATION ACTIVITY 2.2.7: Database system categories.

Match the category to the description.

If unable to drag and drop, refresh the page.

Commercial **NoSQL** **Open source** **Relational**

	MySQL but not MongoDB
	Optimized for big data generated on the internet
	SQL Server but not MySQL
	Allows programmers to inspect and modify source code

Reset

Exploring further:

- DB-Engines Ranking

2.3 Database design and programming

Analysis

A ***database design*** is a specification of database objects such as tables, columns, data types, and indexes. Database design also refers to the process used to develop the specification.

For small, simple databases, the database design process can be informal and unstructured. For large, complex databases, the process has three phases:

1. Analysis
2. Logical design
3. Physical design

The ***analysis*** phase specifies database requirements without regard to a specific database system. Requirements are represented as entities, relationships, and attributes. An entity is a person, place, activity, or thing. A relationship is a link between entities, and an attribute is a descriptive property of an entity.

Terminology

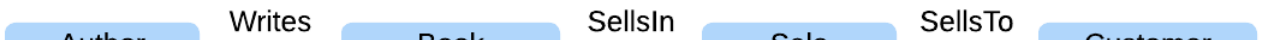
Analysis *has many alternative names, such as conceptual design, entity-relationship modeling, and requirements definition.*

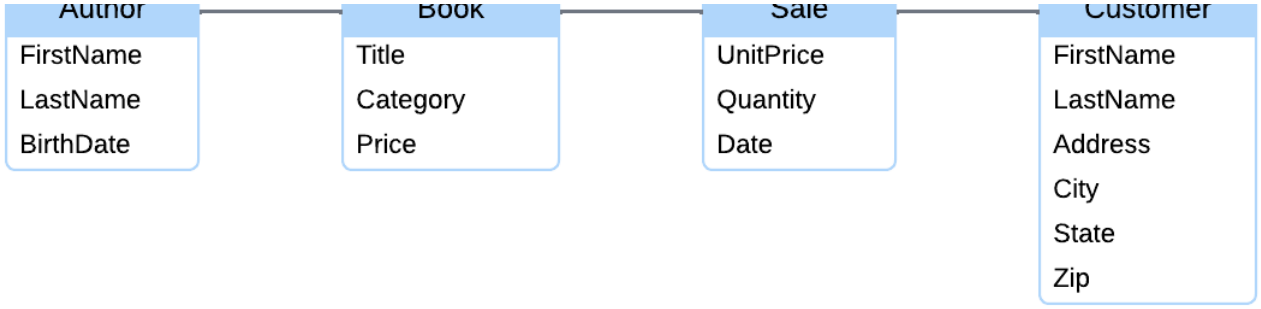
Entities, relationships, and attributes are depicted in ***ER diagrams***:

- Rectangles with round corners represent entities. Entity names appear at the top of rectangles.
- Lines between rectangles represent relationships.
- Text inside rectangles and below entity names represent attributes.

ER diagrams are usually supplemented by textual descriptions of entities, relationships, and attributes.

Figure 2.3.1: ER Diagram.





PARTICIPATION ACTIVITY 2.3.1: Analysis.

Refer to the ER diagram above.

1) What is 'Writes'?
   - Entity
   - Relationship
   - Attribute

2) What is 'Category'?
   - Entity
   - Relationship
   - Attribute

3) What is 'Sale'?
   - Entity
   - Relationship
   - Attribute

4) Is there a direct relationship between Customer and Book?
   - Yes
   - No

Logical design

The ***logical design*** phase implements database requirements in a specific database system. For relational database systems, logical design converts entities, relationships, and attributes into tables, keys, and columns. A ***key*** is a column used to identify individual rows of a table. Tables,

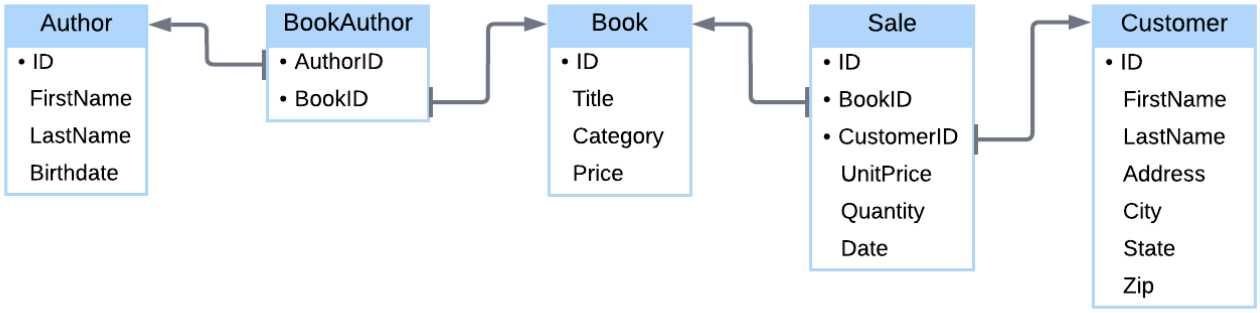
keys, and columns are specified in SQL with CREATE TABLE statements.

The logical design is depicted in a ***table diagram***. Table diagrams are similar to ER diagrams but more detailed:

- Rectangles with square corners represent tables. Table names appear at the top of rectangles.
- Text within rectangles and below table names represents columns.
- Bullets (•) indicate key columns.
- Arrows between tables indicate columns that refer to keys. The tail of the arrow is aligned with the column and the arrow points to the table containing the key.

The logical design, as specified in SQL and depicted in a table diagram, is called a database ***schema***.

Figure 2.3.2: Table Diagram.



PARTICIPATION ACTIVITY 2.3.2: Logical design.

Refer to the ER and table diagrams above.

1) In the table diagram, CustomerID is:

   - ○ A table
   - ○ A key
   - ○ A column that refers to a key

2) In the table diagram, the arrow from Sale to Book corresponds to which column of Sale?

- ID
- BookID
- CustomerID

3) What element of the ER diagram does the BookAuthor table implement?

- Writes
- Book
- Author

4) Logical design is:

- A process only
- A specification only
- Either a process or a specification

5) A schema is depicted in:

- An ER diagram
- A table diagram

Physical design

The ***physical design*** phase adds indexes and specifies how tables are organized on storage media. Ex: Rows of a table may be sorted on the values of a column and stored in sort order. Physical design is specified with SQL statements such as CREATE INDEX and, like logical design, is specific to a database system.

Physical design can be depicted in diagrams. However, logical design is more important for database users and programmers, so physical design diagrams are not commonly used.

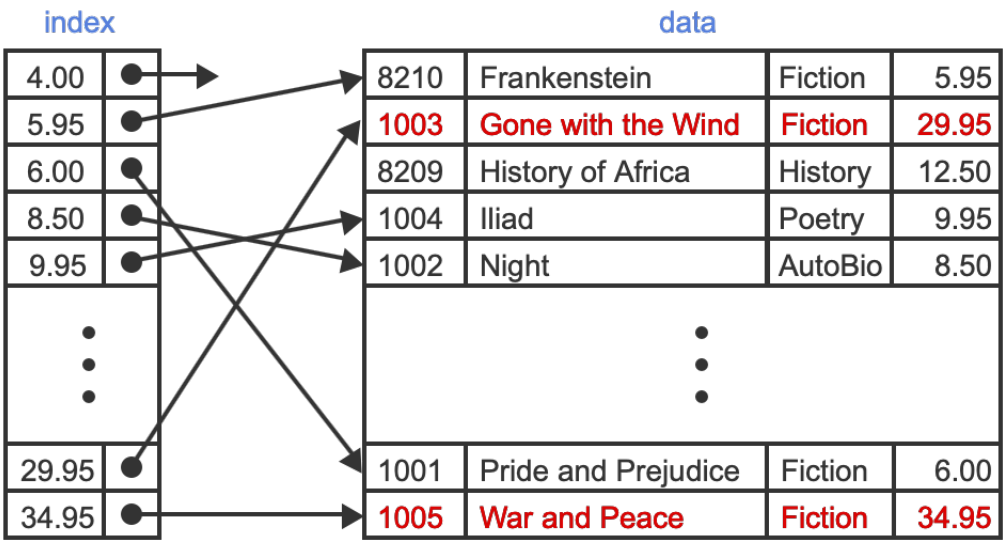
In relational databases, logical and physical design affect queries differently. Logical design affects the query result. Physical design affects query processing speed but never affects the query result. The principle that physical design never affects query results is called ***data independence***.

Data independence allows database designers to tune query performance without changes to application programs. When database designers modify indexes or row order, applications run faster or slower but always generate the same results.

Prior to relational databases, most database systems did not support data independence. Performance tuning often forced time-consuming changes to applications. Data independence is a

major advantage of relational databases and contributed to the rapid adoption of relational technology in the 1980s.

PARTICIPATION ACTIVITY 2.3.3: Data independence.




```
SELECT Title
FROM Book
WHERE Price > 20.0
```

Result

Title
Gone with the Wind
War and Peace

Animation content:

Static figure:
A rectangle on the left has caption Index. A rectangle on the right has caption Data.

Index contains book prices, with two columns and seven rows. The first column has prices 4.00, 5.95, 6.00, 8.50, 9.95, 29.95, and 34.95. The second column has bullets, representing pointers to rows of Data.

Data contains book information, with four columns and seven rows. The first column contains numeric book IDs. The second column contains book titles. The third column contains book categories, such as Fiction and History. The fourth column contains book prices.

An arrow points from each bullet in Index to one row of Data. The price in the row at the arrow tail matches the price in the row at the arrowhead.

This query appears:
Begin SQL code:

```
SELECT Title
FROM Book
WHERE Price > 20.0;
End SQL code.
```

A table named Result appears below the query. The table has one column named Title. The table has two rows containing the titles War and Peace, and Gone with the Wind. In Data, the prices of these two titles are both greater than 20.00.

Step 1: The initial physical design sorts Book rows by ID. The rows of Data are sorted by the values in the ID column.

Step 2: The SQL query selects book titles that cost more than $20.00. Rows of Data are highlighted, one after another. The two books with price > 20.00 appear in the Result table.

Step 3: A new physical design sorts Book rows by Title. Rows of Data are sorted by Title.

Step 4: The new physical design has an index on Price. The Index appears, with arrows to rows of Data. Index rows are ordered by increasing price. Only the last two rows have price > 20.00.

Step 5: The query scans the index rather than the table, and thus is faster. However, Result is the same. Each row of Index is highlighted. When the last two rows are highlighted, the corresponding Data row is highlighted and the title in the row moves to Result.

Animation captions:

1. The initial physical design sorts Book rows by ID.
2. The SQL query selects book titles that cost more than $20.00.
3. A new physical design sorts Book rows by Title.
4. The new physical design has an index on Price.
5. The query scans the index rather than the table, and thus is faster. However the result is the same.

PARTICIPATION ACTIVITY | 2.3.4: Database design process.

Match the term to the description.

If unable to drag and drop, refresh the page.

Logical design | Analysis | Database design | Physical design

	Implementation of database requirements as tables, keys, and columns in a specific database system
	The overall process of determining and implementing database requirements
	Specification of database requirements without regard to implementation
	Affects query performance but not query results

Reset

Programming

Because of data independence, relational database applications can be programmed before the physical design is in place. Applications may run slowly but will generate correct results.

SQL is the standard relational query language but lacks important programming features. Ex: Most SQL implementations are not object-oriented. To write a database program, SQL is usually combined with a general-purpose programming language such as C++, Java, or Python.

To simplify the use of SQL with a general-purpose language, database programs typically use an application programming interface. An ***application programming interface***, or ***API***, is a library of procedures or classes that links a host programming language to a database. The host language calls library procedures, which handle details such as connecting to the database, executing queries, and returning results. Ex: JDBC is a library of Java classes that access relational databases.

Dozens of database APIs are available. Each programming language supports a different API. Major programming languages like C++ and Java support several APIs.

PARTICIPATION ACTIVITY 2.3.5: Database programming with Python.

```
bookCursor = bookDatabaseConnection.cursor()
bookQuery = ('SELECT Title, Category'
             'FROM Book '
             'WHERE Price > 20.00')
bookCursor.execute(bookQuery)

for resultRow in bookCursor.fetchall():
   print('Title:', resultRow[0])
   print('Category:', resultRow[1])
```

Book

• ID	Title	Category	Price
1001	Pride and Prejudice	Fiction	23.00
1002	Night	AutoBio	8.50
1003	Gone with the Wind	Fiction	29.95

```
Title: Pride and Prejudice
Category: Fiction
Title: Gone with the Wind
Category: Fiction
```

Animation content:

Static figure:
Begin Python code:
bookCursor = bookDatabaseConnection.cursor()
bookQuery = ('SELECT Title, Category'
'FROM Book '
'WHERE Price > 20.00')
bookCursor.execute(bookQuery)
for resultRow in bookCursor.fetchall():
print('Title:', resultRow[0])
print('Category:', resultRow[1])
End Python code.

A table named Book has columns ID, Title, Category, and Price. ID has a bullet. The table has three rows with these values:
1001, Pride and Prejudice, Fiction, 23.00
1002, Night, AutoBio, 8.50
1003, Gone with the Wind, Fiction, 29.95

A console image displays the title and category of two books:
Title: Pride and Prejudice
Category: Fiction

Title: Gone with the Wind
Category: Fiction

Step 1: The Book table contains book ID, title, category, and price.

Step 2: The Python code fragment uses the Connector/Python API to access the MySQL database system.

Step 3: A cursor object helps extract query results. The bookCursor object connects to the book database. The first line of code is highlighted.

Step 4: bookQuery contains a SELECT query that selects the title and category for books that cost more than $20.00. Code lines 2 through 4 are highlighted.

Step 5: The execute() method executes the SELECT query. Code line 5 is highlighted.

Step 6: Each pass through the for loop fetches one query result row into the resultRow variable. Code line 6 is highlighted.

Step 7: resultRow has an element for each result column. The print statements print both elements. The last two code lines are highlighted. The titles and categories of two books, with price > 20.00, appear on the console.

Animation captions:

1. The Book table contains book ID, title, category, and price.
2. The Python code fragment uses the Connector/Python API to access the MySQL database system.
3. A cursor object helps extract query results. The bookCursor object connects to the book database.
4. bookQuery contains a SELECT query that selects the title and category for books that cost more than $20.00
5. The execute() method executes the SELECT query.
6. Each pass through the for loop fetches one query result row into the resultRow variable.
7. resultRow has an element for each result column. The print statements print both elements.

PARTICIPATION ACTIVITY 2.3.6: Database programming.

1) Each host language, such as Java or

1) Each host language, such as Java or C++, has a different API.

- True
- False

2) In the animation above, the cursor helped extract individual rows of the query result for processing in a loop.

- True
- False

3) SQL is a general-purpose programming language.

- True
- False