5.2 Storage media

Storage media

Computers use a variety of media to store data, such as random-access memory, magnetic disk, optical disk, and magnetic tape. Computer media vary on four important dimensions:

- Speed. Speed is measured as access time and transfer rate. ***Access time*** is the time required to access the first byte in a read or write operation. ***Transfer rate*** is the speed at which data is read or written, following initial access.
- Cost. Cost typically ranges from pennies to dollars per gigabyte of memory, depending on the media type.
- Capacity. In principle, any media type can store any amount of data. In practice, capacity is limited by cost. Expensive media have limited capacity compared to inexpensive media.
- Volatility. ***Volatile memory*** is memory that is lost when disconnected from power. ***Non-volatile memory*** is retained without power.

Three types of media are most important for database management:

- ***Main memory***, also called ***random-access memory (RAM)***, is the primary memory used when computer programs execute. Main memory is fast, expensive, and has limited capacity.
- ***Flash memory***, also called ***solid-state drive (SSD)***, is less expensive and higher capacity than main memory. Writes to flash memory are much slower than reads, and both are much slower than main memory writes and reads.
- ***Magnetic disk***, also called ***hard-disk drive (HDD)***, is used to store large amounts of data. Magnetic disk is slower, less expensive, and higher capacity than flash memory.

Main memory is volatile. Flash memory and magnetic disk are non-volatile and therefore considered storage media. Databases store data permanently on storage media and transfer data to and from main memory during program execution.

Table 5.2.1: Media characteristics (circa 2018).

Media type	Access time (microseconds)	Transfer rate (gigabyte/second)	Cost ($/gigabyte)	Volatile

Main memory	.01 to .1	> 10	> 1	Yes
Flash memory	20 to 100	.5 to 3	around .25	No
Magnetic disk	5,000 to 10,000	.05 to .2	around .02	No

Feedback?

PARTICIPATION ACTIVITY 5.2.1: Storage media.

Match the media type to the descriptive phrase.

Flash memory	Reading one gigabyte takes about one second. Read time = access time + (transfer rate × gigabytes transferred). Transfer rate from flash memory to main memory ranges from .5 to 3 gigabytes per second. An access time of 20 to 100 microseconds is insignificant compared to transfer time for one gigabyte.	Correct
Magnetic disk	Used to store petabytes (millions of gigabytes) of user data in the cloud. In principle, flash memory can be used to store petabytes of data. In practice, flash memory is too expensive for petabytes of data, so magnetic disk is commonly used.	Correct
Main memory	Upgrading from 16 to 32 gigabytes costs an extra $400 for an Apple laptop computer. An Apple MacBook Pro main memory upgrade from 16 to 32 gigabytes	Correct

costs $400 extra as of October 2019.

Reset

Feedback?

Sectors, pages, and blocks

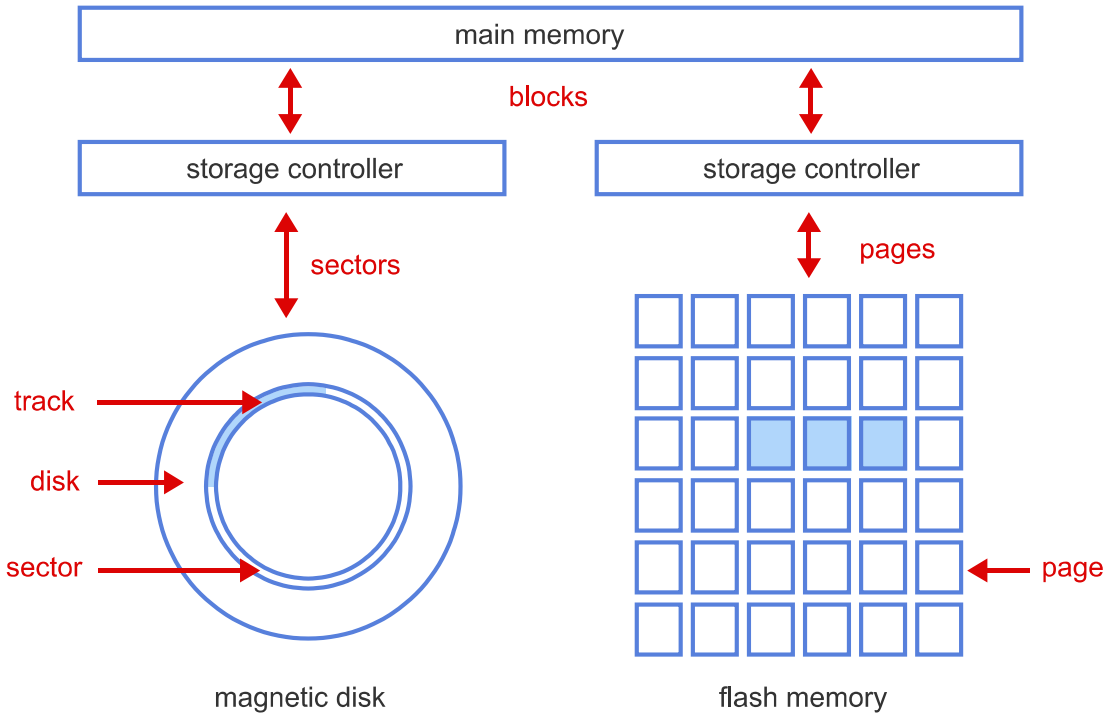
Magnetic disk groups data in ***sectors***, traditionally 512 bytes per sector but 4 kilobytes with newer disk formats. Flash memory groups data in ***pages***, usually between 2 kilobytes and 16 kilobytes per page.

Databases and file systems use a uniform size, called a ***block***, when transferring data between main memory and storage media. Block size is independent of storage media. Storage media are managed by controllers, which convert between blocks and sectors or pages. This conversion is internal to the storage device, so the database is unaware of page and sector sizes.

Block size must be uniform within a database but, in most database systems, can be specified by the database administrator. Database systems typically support block sizes ranging from 2 kilobytes to 64 kilobytes. Smaller block sizes are usually better for transactional applications, which access a few rows per query. Larger block sizes are usually better for analytic applications, which access many rows per query.

PARTICIPATION ACTIVITY 5.2.2: Sectors, pages, and blocks.

1 2 3 4 2x speed



Storage controllers convert blocks to sectors on magnetic disk.

Captions

1. Magnetic disks store data in sectors. Each sector is a segment of a circular track.
2. Flash memory stores data in pages.
3. Storage controllers convert blocks to pages on flash memory.
4. Storage controllers convert blocks to sectors on magnetic disk.

Feedback?

PARTICIPATION ACTIVITY 5.2.3: Sectors, pages, and blocks.

1) Approximately how many sectors are required to store one megabyte of data?

- Always 8,000
- Always 2,000
- Either 250 or 2,000

Correct

If magnetic memory uses 512-byte sectors, storing one megabyte requires 1,000,000 bytes / 512 bytes/sector, or approximately 2000 sectors. With newer four-kilobyte sectors, storing one megabyte requires 1,000,000 / 4,000, or 250 sectors.

2) The database administrator specifies an eight-kilobyte block size. Flash memory page size is two kilobytes. A user runs a query that reads two pages of flash memory. How many blocks are transferred to main memory?

- One-half
- One
- Four

Correct

Data is transferred to the database in blocks. Although four kilobytes are read from flash memory, a minimum of one eight-kilobyte block must be transferred into memory.

Feedback?

Row-oriented storage

Accessing storage media is relatively slow. Since data is transferred to and from storage media in blocks, databases attempt to minimize the number of blocks required for common queries.

Most relational databases are optimized for transactional applications, which often read and write individual rows. To minimize block transfers, relational databases usually store an entire row within one block, which is called ***row-oriented storage***.

Row-oriented storage performs best when row size is small relative to block size, for two reasons:

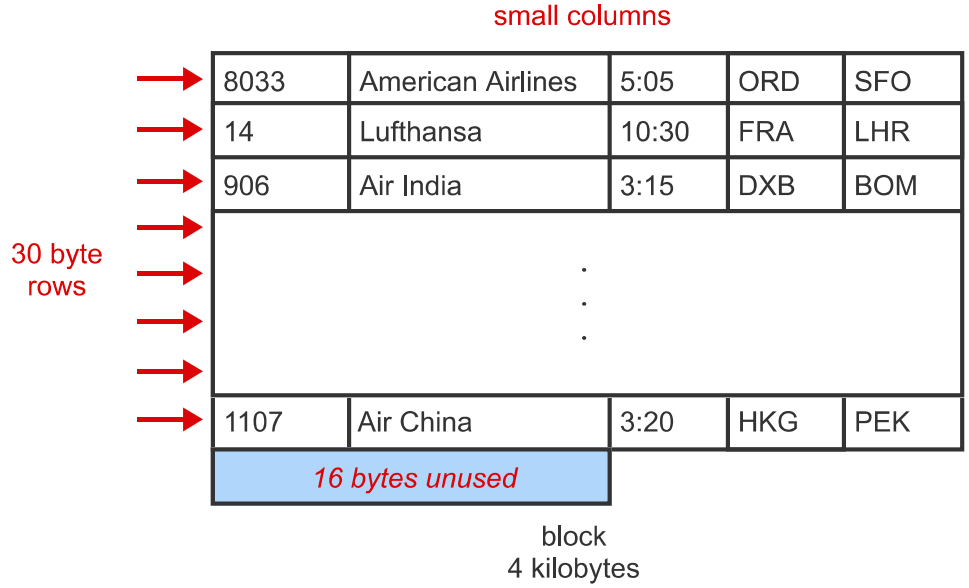
- **Improved query performance.** When row size is small relative to block size, each block contains many rows. Queries that read and write multiple rows transfer fewer blocks, resulting in better performance.
- **Less wasted storage.** Row-oriented storage wastes a few bytes per block, since rows do not usually fit evenly into the available space. The wasted space is less than the row size. If row size is small relative to block size, this wasted space is insignificant.

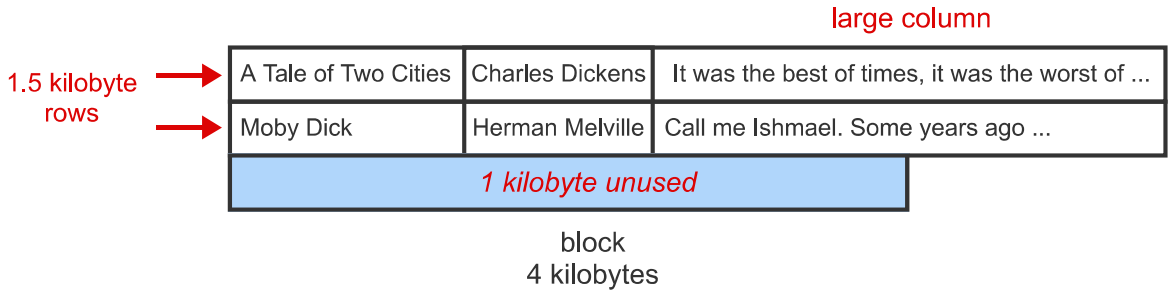
Consequently, database administrators might specify a larger block size for databases containing larger rows.

Sometimes a table contains a very large column, such as 1 megabyte documents or 10 megabyte images. For tables with large columns, each row usually contains a link to the large column, which is stored in a different area. The large column might be stored in files managed by the operating system or in a special storage area managed by the database. This approach keeps row size small and improves performance of queries that do not access the large column.

PARTICIPATION ACTIVITY 5.2.4: Row-oriented storage.

1 2 3 4 2x speed





One kilobyte goes unused, so wasted space is significant.

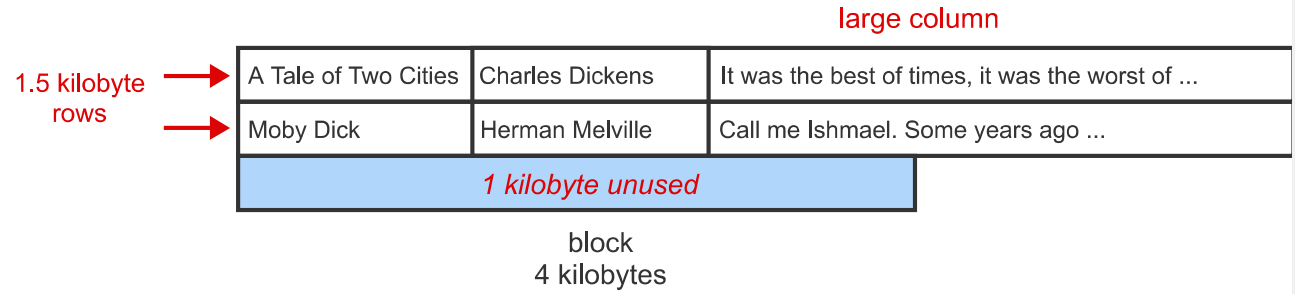
Captions

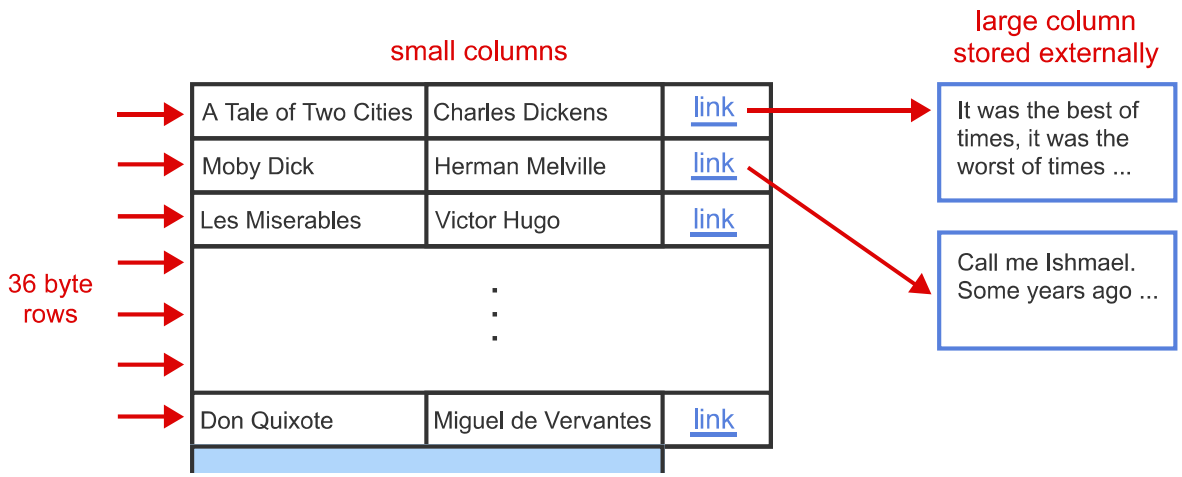
1. The row size is small relative to the block size, so many rows are transferred per block.
2. Only 16 bytes go unused, so wasted space is insignificant.
3. The row size is large relative to block size, so fewer rows are transferred per block.
4. One kilobyte goes unused, so wasted space is significant.

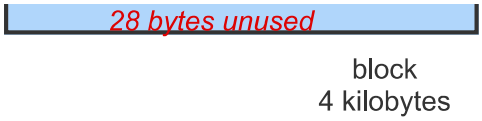
Feedback?

PARTICIPATION ACTIVITY 5.2.5: Row-oriented storage with large columns.

1 2 3 2x speed







More rows fit per block, and less space is wasted.
Queries that do not access the large column are faster.

Captions

1. The large column allows only two rows to be transferred per block and wastes significant space.
2. Large columns are replaced by a link and are stored in a separate area.
3. More rows fit per block, and less space is wasted. Queries that do not access the large column are faster.

Feedback?

PARTICIPATION ACTIVITY 5.2.6: Row-oriented storage.

1) A database uses 16-kilobyte blocks. A table contains 18,200 rows of 100 bytes each. Ignoring space used by the system for overhead, how many bytes are unused on each block? Assume one kilobyte = 1,024 bytes.

- None
- 84
- 384

Correct

16 kilobytes = 1,024 × 16 = 16,384 bytes. Since rows are not split across blocks, 16,384 bytes / 100 bytes/row = 163 rows are stored on each block. The number of unused bytes = 16,384 bytes - (163 rows × 100 bytes/row) = 84 bytes.

2) 4-megabyte photographs are usually stored ____.

- in the same blocks as other columns of the table
- separate from other columns,

Correct

Database management of large columns enables database services, such as security and backup, on the large columns. In some cases, however, the database stores a link to large column data managed by the file system.

managed by the file system

- separate from other columns, managed either within the database or by the file system

Feedback?

Column-oriented storage

Some newer relational databases are optimized for analytic applications rather than transactional applications. Analytic applications often read just a few columns from many rows. In this case, column-oriented storage is optimal. In ***column-oriented*** storage, also called ***columnar storage***, each block stores values for a single column only.

Column-oriented storage benefits analytic applications in several ways:

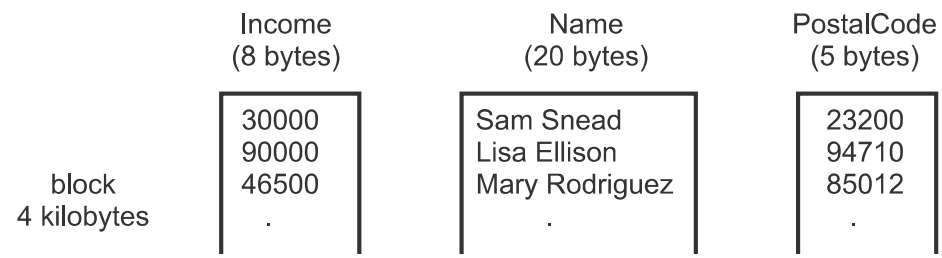
- Faster data access. More column values are transferred per block, reducing time to access storage media.
- Better data compression. Databases often apply data compression algorithms when storing data. Data compression is usually more effective when all values have the same data type. As a result, more values are stored per block, which reduces storage and access time.

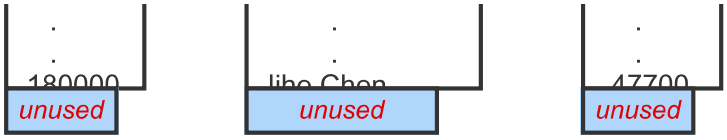
With column-oriented storage, reading or writing an entire row requires accessing multiple blocks. Consequently, column-oriented storage is a poor design for most transactional applications.

PostgreSQL and Vertica are examples of relational databases that support column-oriented storage. Many NoSQL databases, described elsewhere in this material, are optimized for analytic applications and use column-oriented storage.

PARTICIPATION ACTIVITY 5.2.7: Column-oriented storage.

1 2 3 4 2x speed






```
SELECT AVG(Income)
FROM Person
```

AVG(Income)
52844

```
SELECT Name, PostalCode, Income
FROM Person
WHERE PostalCode = 23200
```

Name	PostalCode	Income
Sam Snead	23200	30000

Selecting multiple columns reads multiple blocks and is slower than row-oriented storage.

Captions

1. With column-oriented storage, a 4 kilobyte block contains roughly 500 8-byte column values.
2. Computing the average of all incomes reads fewer blocks than row-oriented storage.
3. Different columns occupy separate blocks.
4. Selecting multiple columns reads multiple blocks and is slower than row-oriented storage.

Feedback?

Terminology

*In this material, the terms **column-oriented** and **columnar** refer to a data storage technique. Occasionally, these terms refer to a type of NoSQL database, commonly called a **wide column database** and described elsewhere in this material.*

PARTICIPATION ACTIVITY 5.2.8: Column-oriented storage.

Refer to the animation above. Assume the database contains 1,000,000 people.

1)

Assume column-oriented storage. Roughly how many blocks are required to compute average income by postal code for all people?

- ○ 2,000 blocks
- ◉ 3,250 blocks
- ○ The number of blocks depends on row size.

Correct

The query must select the Income and PostalCode columns. A 4-kilobyte block contains 4,000 bytes / 8 bytes = 500 income values, so 1,000,000 income values occupy 1,000,000 / 500 = 2,000 blocks. A 4-kilobyte block contains 4,000 / 5 = 800 postal codes, so 1,000,000 postal codes requires 1,000,000 / 800 = 1,250 blocks. 2,000 + 1,250 = 3,250 blocks.

2) Assume row-oriented storage, with each row containing Name, Income, and PostalCode only. Roughly how many blocks are required to compute average income by postal code for all people?

- ○ 3,250 blocks
- ○ 4,133 blocks
- ◉ 8,264 blocks

Correct

Each row requires 33 bytes (8 bytes for Income + 20 bytes for Name + 5 bytes for PostalCode). Roughly 4,000 bytes / 33 bytes = 121 rows fit in a 4-kilobyte block. 1,000,000 people / 121 people/block = roughly 8,264 blocks.

3) The term 'column-oriented' ____.

- ○ always means a technique for organizing data on storage media
- ○ always means queries that specify multiple columns in the SELECT clause
- ◉ means either a technique for organizing data on storage media or a type of NoSQL database

Correct

Usually 'column-oriented' and 'columnar' mean a technique for organizing data on storage media. Sometimes these terms mean a type of NoSQL database, commonly called 'wide column' database.

Feedback?

Exploring further:

- PostgreSQL column-oriented storage
- Vertica column-oriented storage

How was this section? Provide feedback