**CS60 Chapter 01. Introduction to files, databases, and**

**database management systems**

**Contents**

[Introduction 1](#_Toc190509274)

[Computer files and their storage 2](#_Toc190509275)

[Reliability of a disk drive 7](#_Toc190509276)

[Risks to drives and computers from natural disasters, viruses, and theft 7](#_Toc190509277)

[Drive performance 8](#_Toc190509278)

[Mirrored drive arrays and striped drive arrays 8](#_Toc190509279)

[Backing up and restoring data and programs 9](#_Toc190509280)

[Drive maintenance 10](#_Toc190509281)

[Database, table, column, row, entity 10](#_Toc190509282)

[Structured Query Language, SQL 15](#_Toc190509283)

[Oracle's PL/SQL language 15](#_Toc190509284)

[The parts of a database management system 16](#_Toc190509285)

[Steps to design a relational database 17](#_Toc190509286)

[Terminal networks (optional) 18](#_Toc190509287)

[File Server 19](#_Toc190509288)

[Client/Server Computing 19](#_Toc190509289)

[Key Terms 19](#_Toc190509290)

# Introduction

This chapter

• introduces CS60 and some words that we'll use in CS60,

• reviews characteristics of files and disk operations

• summarizes how data

(1) is stored on (written to) magnetic and optical media,

(2) is retrieved from (read from) that media, and

(3) can be lost.

Being aware of these will help motivate you to protect your data and improve database performance (diminish the time necessary to retrieve and store data).

• introduces databases and database management systems.

A programmer need not be an expert in how data is stored and retrieved and how a disk drive works to develop a working relationship with hardware. Neither must a truck driver be capable of designing an engine or be able to repair an engine to have a sense of how to use wisely the resource, not abuse the engine and transmission, and select the right engine for the task.

# Computer files and their storage

**Computer files** on secondary storage are instructions or data with filenames, dates, times, and other properties written and read by an operating system. The operating system works closely with the disk controller. Examples of operating systems are Windows Vista, Windows XP, Macintosh OS 9.0; Unix; Linux; MS-DOS and DR-DOS; VMS, and 1000s of other operating systems under development, presently active, or obsolete. An operating system is precisely referred to by **its name and version or release number** that adds enhancements and fixes bugs. Major operating systems are modified weekly or monthly—far more often than the public hears about new versions such as Windows 7.

On **magnetic secondary storage** (floppy disks, hard drives, cartridge disks, and digital tapes), physically the files are patterns of magnetic fields, essentially little bar magnets of differing polarity (different arrangements of the North and South magnetic poles). These arrangements of bar magnets comprise the 0's and 1's of the binary number system. One of the simplest arrangements, called the **encoding pattern**, that stores the 0's and 1's is shown below:

***Polarity reverses between bits***

**Band 2 Band 1 Band 2 Band 1 Band 2 Band 1 Band 2 Band 1**

**S N N S S N N S S N S N N S S N**

**4th bit, a “1” 3rd bit, a “1” 2nd bit, a “0” 1st bit, a “1”**

**because because**

**band 2 has band 2 has**

**same polarity different polarity**

**as band 1 than band 1**

On **optical secondary storage** (CD- and DVD-ROM optical disks), physically the files are **lands** (reflective surfaces) and **pits or craters** that scatter a laser beam or partially absorb it so the laser intensity sensed by a detector is reduced. Some CD-and DVD-RW (ReWritable) disks use materials that are either crystalline or non-crystalline (amorphous). Rather than using pits to scatter the laser beam and reduce the intensity detected, CD-RW disks use amorphous regions with lower reflectivity to reduce the intensity of the reflected laser. To rewrite to these disks, the absorptive amorphous regions are converted back to reflective crystals.

Some DVD disks have multiple layers on each side where data is recorded. Eventually, DVD and Blu-Ray drives will read data from both sides of a disk without turning over the disk. DVD lasers have shorter wavelengths (in the red part of the visible spectrum rather than the infrared spectrum for CD's), and higher densities (more pits per square centimeter) yielding disk capacities up to 17 GB (gigabytes) rather than 640-700 MB (megabytes) for CD's. Blu-Ray drives use blue lasers with even shorter wavelengths and higher densities.

Secondary storage is **nonvolatile**—the data and instructions remain after the computer is turned off, in contrast to the volatile storage of the semiconductor memory comprising the primary memory (usually dynamic RAM) and cache memories (with static RAM) of most computers. A more accurate adjective than *nonvolatile* is **semipermanent**, because disks and drives will fail. The reliability of drives is discussed in the next section.

Per megabyte of storage, secondary storage is cheaper than the semiconductor memory of primary memory (random access memory or RAM), but secondary storage reads and writes the data much slower than primary memory. If semiconductor memory is battery powered, it too can be used as secondary storage that emulates a disk. This storage is much more expensive per gigabyte than magnetic storage, but also much faster. Furthermore, technologies exist such as flash memory that are nonvolatile. To reduce the time to access data, some databases store frequently accessed data in RAM as well as disk. Changes to the data are written back to disk.

**Access time decreases** to the right

(e.g., the time required to access the data stored in memory is less than the time to access the data stored on a hard drive)

**Cost per megabyte increases** to the right

**Secondary storage** **Memory** **L2 Cache** **L1 Cache Registers**

e.g., hard drive, e.g., dynamic Usually Part of the Part of the

floppy, CD- and RAM; separate from CPU, CPU;

DVD-ROM, flash. volatile CPU; volatile volatile volatile

nonvolatile

The operating system maintains a **directory of files**—a table of contents of the disk—including the filenames, dates, times, and other file attributes. This directory also stores the cluster number where each file begins. A **cluster** is the smallest unit of a file and is composed of one or more **sectors of a track** on the disk. The drive or partition has clusters numbered 0, 1, 2, 3, …, maximum number of clusters. This maximum number depends on how many bits every entry in the **file allocation table** (FAT) can store. The FAT was 16 bits, but for Windows 98, NT, and later versions, the FAT is 32 bits.

The biggest whole number that can be stored in 16 bits is the binary number 11111111 11111111 or 65,53510, so that plus one (or 65536 since 00000000 00000000 also can be stored) is the maximum number of clusters possible in that partition if 16 bits are available for numbering the clusters. A 32-bit FAT could theoretically store over four billion clusters.

A 16- or 32-bit storage area is available in the FAT for every cluster. In that storage area is stored one of four things:

(a) whether that cluster is unused and available for writing

(b) the cluster number of the next cluster for the file (if there is another cluster in the file)

(c) whether this cluster is the end of a file

(d) whether this cluster is bad and cannot be used to store data, as determined from manufacturing tests or low-level formatting of the drive

Thus the directory points to the first cluster number of a file, and at that cluster number in the FAT is stored the cluster number of the next cluster of the file, and so on until the end of the file is reached. If the file size of an existing file has to grow, then the marker in the FAT that indicates that this was the last cluster is replaced with the cluster number of an added cluster, and so on again until the last new cluster is reached which again records that it is the last cluster of the file. As file sizes grow and new files are added, operating systems use various strategies to select from amongst the available (unused, good) clusters.

**Directory of a partition:**

**Filename and extension Date, time, other file attributes FAT# of first cluster**

|  |  |  |
| --- | --- | --- |
| earlier files |  |  |
| CS60 Chapter 00.doc |  | 00000000 000001002 = 410 |
| other files |  |  |
| CS60 Chapter 01.doc |  | 00000000 001010102 = 4210 |
| still other files |  |  |

**File Allocation Table for a partition:**

**FAT# in decimal Entry giving address for next cluster, End-of-File, Unused, or Bad**

|  |  |
| --- | --- |
| 1 |  |
| 2 |  |
| 3 |  |
| 4 | 00000000 000001012 = 510  (the next cluster of this file is #5) |
| 5 | 00000000 000001102 = 610  (the next cluster of this file is #6) |
| 6 | 00000000 000001112 = 710  (the next cluster of this file is #7) |
| 7 | End-Of-File[[1]](#footnote-2) (this is the last cluster of this file) |
| 8 | Unused and available |
| 9 | Unused and available |
| 10 | Bad cluster, unavailable |
| ... |  |
| 41 | Unused and available |
| 42 | 00000000 001010112 = 4310  (the next cluster of this file is #43) |
| 43 | End-Of-File (this is the last cluster of this file) |
| 44 |  |
| ... |  |
| 65535  (for a 16-bit FAT) |  |

*16 or 32 bits are available here to store the cluster number of the next cluster in that file or one of three codes: a code that indicates that this cluster is that last one in the file, a code that indicates that that cluster is unused and available, or that cluster is bad and cannot be used.*

A drive can be divided into several **partitions** or one partition can entirely fill the drive. For example, one physical drive can be partitioned into logical drives C, D, and more. Each partition has its own directory and FAT.

The hardware controller for the drive moves the heads to the proper radius and reads the data from or writes the data to the proper surface of a multi-disk drive. While 3.5-inch floppy drives rotate at 5 revolutions per second (300 rpm), hard drives read and write faster because they rotate at 90 revolutions per second (5400 rpm) to 250 revolutions per second (15,000 rpm). Hard drives also have their own memory buffers of 128Kbytes (1K = 1024) to 8MB (8 megabytes where a megabyte is 10242 = 1,048,576 bytes or about 1.05 million bytes) to store data just read or about to be written.

Since cluster sizes are typically 512 bytes to 4K bytes or substantially larger (even 32K or 64K with 16-bit FATs and gigabyte-sized partitions), databases usually require many clusters to store the data.

For a file that is being written to a disk, the process of **closing** the file includes writing any data remaining in the write buffer to the disk, writing an **end-of-file (EOF)** marker at the end of the data, filling out the rest of the cluster with some character, and completing the entries in the disk directory and FAT.

The clusters of a file need not be **contiguous (**neighboring clusters on the same track, adjacent tracks on the same platter, or the same cylinder on different platters**)**. If the first cluster of a file is, say, Cluster # 10000 on the disk, the next cluster of the file need not be Cluster #10001. Clusters may be scattered across the drive on tracks of different radii and different surfaces of a multi-platter drive. If the clusters are not contiguous, then the file is **fragmented**. A **defragmenting utility** can be used to move the data so it’s contiguous, although not without risk. An ideal situation would be that the file fill the drive cylinder by cylinder, where a **cylinder** is the tracks with the same radius on different disks. In this way, the heads don’t have to move as much as it would if the strategy were to fill all tracks of one surface, then begin to fill another surface. All heads are rigidly connected to an arm and driven by one motor, so at each instant the heads for the different platters are on the same track with the same radius.

**File Allocation Table with a fragmented file:**

**Fat# Entry giving address for next cluster,**

**End-of-File, Unused, or Bad**

|  |  |
| --- | --- |
| 1 |  |
| 2 |  |
| 3 |  |
| 4 | 00000000 000001102 = 610 (This fragmented file skips to cluster 6) |
| 5 | Unused (might also be used by another file) |
| 6 | End-Of-File (The file ends with this cluster) |
| 7 | Bad |
| 8 |  |
| ... |  |

Several issues are of special importance with databases because they can require extensive reading from and writing to the secondary storage and because of the value of the data:

 reliability of the drive

 other risks to the drive and computer such as natural disasters and theft

 drive performance

 mirroring and striping of drives

 backing up and restoring data and programs

 drive maintenance

These are introduced in the following sections.

# Reliability of a disk drive

Drives are electronic and mechanical devices with chips, a motor to turn the disks, another motor to move the arms to which the read/write heads are attached, bearings for the motors, arms that support the heads, and the disk platters. Their lifetimes are finite. They have theoretically calculated **Mean Times between Failures** (**MTBF**s) and **warranty periods** (the time periods after purchase during which the manufacturer or distributor will replace the drive if it fails). The warranty period reflects the manufacturer’s belief that few drives will fail during that period because of the high cost to the manufacturer to replace a drive. All drives fail at some time. The issues are when and what losses you will incur when they do fail.

Drives also have **error rates** associated with writing and reading data.

Drive specifications include acceptable temperature and pressure (or altitude) ranges for their operation because the temperature and pressure of the ambient air will affect the drive’s temperature. Some drives require fans to blow air directly on them to operate properly. Hot and freezing cars and warehouses and the altitudes of robots can cause temperatures and pressures to fall outside acceptable ranges.

# Risks to drives and computers from natural disasters, viruses, and theft

Natural disasters include

 Fires

 Floods and other water damage from raining through windows and leaks above the computer

 Computer overheating because the ambient temperature is too high, pressure is too low, cooling fans fail, vents are blocked, or accumulated dust on a drive blocks heat from escaping.

 Computer too cold.

 Earthquakes

 Computer racks and cases falling over and shelves breaking

 Power outages or brownouts, voltage spikes and lightning strikes fed into the computer through modem and network connections as well as the power cable

Disasters created by man and woman include

 Viruses, worms, and Trojan horses that damage or delete data and programs, prevent access to the data or programs, block the writing of data to the disk, or fill up the drive with garbage so no room remains for a file size to grow or a new file to be written to that drive.

 Deliberate sabotage by competitors, disgruntled employees, malicious criminals, and hackers

 Theft of drives, computers, and backups

 Theft of data for espionage or sale to commercial competitors.

# Drive performance

What data transfer rate is possible while reading from and writing to a drive? What are the rates in **burst mode** (the rate to transfer a relatively small amount of data) and in a **sustained mode** (the rate to transfer large amounts of data, a rate that can be continued indefinitely)?

These rates are limited by

 the transfers from the drive buffer to either the CPU or directly to primary memory (with Direct Memory Addressing (DMA)), or to the drive buffer from the CPU or primary memory

 the transfers to and from the rotating disk and movements of the heads from track to track? This sustained rate depends on the rotational rate of the disk and the number of sectors per track (which may depend on the radius or track number). This rotational delay is called the **drive latency**. It also depends on the time for the head to move to another track (**the seek time**) and any **settling time** to dampen out vibrations.

The detailed arrangement of the data can influence the actual transfer rate because clusters scattered all over the disk would influence how long the head must wait for the data to rotate around and pass under the head. The times to move the head between tracks depend on the number of head moves and what tracks are involved (essentially the distances between the tracks where the data lies). Fragmentation can significantly affect performance.

# Mirrored drive arrays and striped drive arrays

**Data mirroring** means simultaneous writing of the same data to two or more drives. If a drive fails, then the remaining drive(s) store the data (unless the cause of failure wipes out all drives). For two drives, the cost doubles for the drives to store the same data, and the cost is greater for a more sophisticated drive controller capable of mirroring.

Write times for mirroring would be the same as for a single drive, but read times are nearly halved because half of the data could be read from one drive and half from the other.

**Hot swapping** of a bad drive (replacing a bad drive without shutting down the computer and without disrupting computer operations) is possible, so the computer can continue operating while the bad drive is replaced with a good one, and the data is then automatically copied to the new drive. If the cause of failure is that some media has become bad rather than complete drive failure, the controller and software will automatically write the data to good clusters.

**Striped drive arrays** require three or more drives. If three drives are used, the last drive is used for error checking (e.g., writing parity bits). For three or more drives, the data is partitioned between the drives, and the check drive tests the validity of the data on the other drives. If any one drive fails in a 3-drive array, no data is lost. Writing time can be reduced since each drive writes only part of the data. Reading time can be reduced since several drives can simultaneously supply the data.

Mirrored and striped drive arrays are forms of **RAID,** an acronym for a poorly named but valuable and practical concept: **Random Array of Inexpensive Drives**. With increasing levels of RAID, multiple drives can simultaneously fail without losing data.

# Backing up and restoring data and programs[[2]](#footnote-3)

The value of the data and the importance of continued access after disasters means that the database (the files of data), Database Management System (DBMS) and other applications, and system software must be backed up regularly and systematically, the backups safeguarded, and the backups be able to restore the operating system, programs, and data. If a fire or other disaster to the computer and drives also wipe out the backups, then the organization still has lost the data no matter how carefully the data was backed up.

Backing up use to mean writing to a tape all operating system, programs, and data (**full backups**). With a series of numbered tapes, full backups were followed by **incremental backups** that stored any changes to the files, new files, and what files were deleted. Restoring would begin with restoring the complete backup followed in order by the incremental backups that would delete, replace, and add files.

Backing up now can also use media other than magnetic tape, including

 CD-R (CD’s that can be recorded once),

 CD-RW (rewritable CD’s), and

 DVD-R-, DVD-R+, DVD-RW, DVD-RAM (writable or rewritable DVD optical disks)

 cartridge disk drives (e.g., ZIP), and even

 regular IDE (Integrated Drive Electronics) and SCSI (Small Computer System Interface) drives mounted in removable drawers with quick disconnects so the drives can be removed and stored safely.

Within the limits of 1.44MB, a floppy disk can back up selected files. Flash memory connecting to USB ports is becoming increasingly popular, but writing to flash memory is relatively slow. While flash memory has essentially unlimited reads, it allows only a finite number of writes to the same memory cells (maybe 100,000 to a million times).

Just as the secondary storage must be reliable, the backup storage also must be reliable.

# Drive maintenance

Drive maintenance includes occasional checking on drive fragmentation, defragmenting the drive if warranted, checking for any lost segments of the files (where the number in the FAT pointing to the next cluster of a file is corrupted or wrong), and testing the media surface to see if it can store data.

Computers need periodic cleaning. Considerable dust can settle over the electronics and drives, which blocks cooling.

# Database, table, column, row, entity

A **database is an organized collection of related information stored in a file.** A database for a relational database management system such as Oracle, SQL Server, MySQL, and Microsoft Access (mentioned because of their courses at Santa Monica College) has one or more tables, each with one or more columns and rows. A **column** is a synonym for a **field** or **attribute**. A **row** is a synonym for a **record**. A field also can be the value stored at the intersection of a row and column. An **entity** is a table or a group of tables.

Here’s a table in Oracle format:

**REPRESENTATIVE table**

| **REPRESENTATIVE\_ID** | **LAST\_NAME** | **FIRST\_NAME** | **REGION** | **HIRE\_DATE** | **PHONE** |
| --- | --- | --- | --- | --- | --- |
| 11 | Rogler | Harold | SW | 05-JAN-99 | (310) 456-7890 |
| 22 | Higgins | Heather | SE | 16-DEC-91 | (404) 524-8472 |
| 33 | Sullivan | Pat | NE | 21-FEB-88 | (305) 734-2987 |
| 44 | Speed | Kristen | MW | 14-JUN-90 | (708) 823-8222 |
| 55 | Sigafoos | Alex | NW | 05-MAR-01 | (310) 123-7890 |

The table has a name, REPRESENTATIVE, and the six vertical columns each have names (case insensitive in Oracle) and each column stores a particular kind of data[[3]](#footnote-4). These names are carefully selected to describe the table or column, just as you carefully selected meaningful and descriptive names for variables, constants, user-defined functions, procedures, and objects such as command buttons in the computer languages you’ve studied. These columns store the properties or attributes of one of the entities.

The raw data in this table is presented in five horizontal rows, a row for each representative. Already you can see that the database is more than the values in those rows, for the table and column names are part of the database. Also part of the database are the datatype for each column—four columns of text with columns two and three limited to 20 characters and column four with two characters, a column of dates, and another column of text. While the first column appears here as integers, actually they're being stored digit by digit as a string. The values for the REPRESENTATIVE\_ID are not just any digits. They are unique and lie in the range of (say) 01 to 99. Their uniqueness and their range are enforced during data entry by **data validation rules.** These rules are also stored as part of the database.

I’ll next review simple ideas on how data is stored in a file (even a text file)—ideas that you’ve perhaps used in some programming course.

Sometimes files of data include special characters called **delimiters** that separate the various fields and various records. Delimiters are often commas, semicolons, tabs, or quotation marks between fields and a carriage return or combination carriage return/line feed (ASCII characters 13 and 10) between records.

Comma-delimited files or files with other delimiters can be **exported** from a DBMS (a Database Management System program such as Oracle, SQL Server, Access, or MySQL) as a text file. **Text files** are files of ASCII characters (or other collating sequences such as Unicode) that can be viewed, edited, and saved from an ordinary text editor such as Notepad. But note that other data exists in a database other than this data output to a text file—data such as table names and column names, datatypes designating how data will be stored in a column, data validation rules—so information can be lost in an export.

For some databases, the delimiters allow the fields to have different lengths in different records. Such a database can have **variable-length records**. In fact, a field can be empty. For example, if the field delimiter is a <tab>, then <tab><tab> indicates that no information is stored for the field.

Some files of data have a fixed structure. If the corresponding fields of various records have the same lengths, then that table has **fixed-length records**. In such a database, no delimiters between fields or records are necessary. If the size of the fields are fixed and known, then the computer can calculate an **offset** (perhaps measured in bytes) from the first record to reach any other record or field.

To search sequentially through a variable-record-length database can be time consuming because of the many tests for delimiters necessary to determine which field of which record is being accessed. But most of this run-time burden can be eliminated by **indexing** the list of data so the testing for the delimiters and determining the addresses of the various fields and records is done once and the results stored in a table. Thereafter, the index is accessed to determine the address of each field (or at least the address of each record). The index table can also pre-record any arrangement (ascending or descending alphabetical or numerical value) of any field so searching and sorting records based on that field is much faster.

Another characteristic of databases is how the data is stored. One way is to store every letter, decimal digit, punctuation mark, or other character as an ASCII character or some other

collating sequence in one byte or two bytes[[4]](#footnote-5) each. This approach requires that conversions take place between the ASCII characters for numbers and the binary forms used internally in the computer for arithmetic.

Even text need not be stored as one byte per character, for it may have been **compressed** by one of many schemes. One compression scheme is to assign a number in the range 0 to 65,535 to the 65,536 most popular words. If the word *hippopotamus* were assigned the number 4096, that integer could be stored in the two bytes with bits 00010000 00000000 rather than 12 separate bytes for *hippopotamus.*

Another way to store numbers is the binary form native for that data, where an integer in the range –32768 to 32767 is stored in its two-byte (16-bit) binary form, a floating point number such as -0.12345x104 is stored in its binary form with parts of the storage used for the sign (-), mantissa (0.12345), and exponent (4).

Of course, information includes much more than the text and numbers we read from the printed page. Faxes, sounds, still and animated graphical images, and digital video each have different formats. The binary data of a movie in digital format can be a field. With a binary field as large as four gigabytes, one movie can be one field in an Oracle database. More commonly, such huge fields are stored outside the database under the control of the operating system, but with pathnames (drive, folders, and filenames) to these external files stored inside the database.

Access to the data can be sequential or random. **Sequential access** means that the database file is accessed record by record beginning with the first. **Random access** means that the records can be accessed in any order or it can be accessed sequentially also since the order 1, 2, 3, … is a possible order of access.

In the case of a file being sequentially accessed with some 3rd Generational Language, you read each record by reading a series of variables (the fields that comprised the record). Fields are separated by a comma (or another delimiter) and records are separated by a <Carriage Return> (or another end-of-line delimiter). Any strings begin and end with a double-quote mark ("). Conversions between the native data types as stored in memory and the ASCII data in the file were automatic.

In the case of a random access file, you write and read records in random order in the native binary format of the data (without conversions necessary between the native format and what was written in the disk). After a record was read, you sort out the fields by using the RecordName.FieldName reference.

With many computer languages, you can retrieve data from relational databases such as Oracle.

Oracle, IBM DB2, Microsoft SQL Server, and the popular PC databases such as dBase, FoxPro, Paradox, and Access are **relational database management systems (RDBMS)**. The precise definition of a RDBMS is based on relational mathematics, and we will study its rules later. For now, a RDBMS can link (join) two or more tables so data can be extracted from one or more tables. Each table is composed of columns and rows. A row is a record, and a record has one or more fields. For now, a relation is a special table, one that has a unique name in the database, each column has a unique name in the table it is part of, and each column has a datatype. There are other rules we’ll see later.

A relational database can link (**join**) two or more tables if the tables are properly designed. What links two tables is that a column (or maybe more than one column) in a table references or links to a column (or more than one column) in a table. For example on the next page are four tables modified from Capron, 6th edition, with lists of the orders, customers, inventory, and sales representatives.

The first two tables (INVOICE and CUSTOMER) are linked by the CUSTOMER\_NUMBER field, the first and third tables (INVOICE and INVENTORY) are linked through the ITEM\_NUMBER field, and the 2nd and 4th tables (CUSTOMER and REPRESENTATIVE) are linked through the REPRESENTATIVE\_ID field.

The four tables are not separate files but are integrated into one file perhaps along with other pieces such as instructions to generate reports, queries, forms, macros, etc. The data and instructions are manipulated with the database management system (DBMS).

**INVOICE** table

| **INVOICE\_**  **NUMBER** | **CUSTOMER\_**  **NUMBER** | **INVOICE\_**  **DATE** | **ITEM\_**  **NUMBER** | **QUANTITY** |
| --- | --- | --- | --- | --- |
| 01 | 20 | 12-MAY-99 | 70 | 11 |
| 02 | 30 | 28-FEB-99 | 60 | 15 |
| 03 | 30 | 13-SEP-00 | 20 | 14 |
| 04 | 20 | 10-JUL-01 | 10 | 10 |
| 05 | 60 | 31-AUG-01 | 60 | 20 |
| Primary key | Foreign key |  | Foreign key |  |
| Varchar2(2) | Varchar2(2), NOT NULL | Date | Varchar2(2) | Number(2) |

**CUSTOMER** table

| **CUSTOMER\_**  **NUMBER** | **CUSTOMER\_NAME** | **CITY** | **REPRESENTATIVE\_ID** |
| --- | --- | --- | --- |
| 10 | Ballard Computer | Seattle | 55 |
| 20 | Computer City | Miami | 33 |
| 30 | Under\_Score, Inc. | Atlanta | 22 |
| 40 | Varner User System | Naperville | 44 |
| 50 | 100% Jargon | Spokane | 55 |
| 60 | Computing Solutions | Tucson | 11 |
| Primary key |  |  | Foreign key |
| Varchar2(2) | Varchar2(20), NOT NULL | Varchar2(20) | Varchar2(2) |

**INVENTORY** table

| **ITEM\_NUMBER** | **DESCRIPTION** | **QUANTITY\_ON\_HAND** |
| --- | --- | --- |
| 10 | Hand Scanner | 191 |
| 20 | Modem | 453 |
| 30 | Hard Drive | 294 |
| 40 | Printer pack | 676 |
| 50 | CD-ROM drive | 817 |
| 60 | 3 1/2" disk holder | 982 |
| 70 | Sound card | 0 |
| 80 | Mouse | 296 |
| 90 | Rogler’s DSL | 152 |
| Primary key |  |  |
| Varchar2(2) | Varchar2(20) | Number(3), NOT NULL |

**REPRESENTATIVE table**

| **REPRESENTATIVE\_ID** | **LAST\_NAME** | **FIRST\_NAME** | **REGION** | **HIRE\_DATE** | **PHONE** |
| --- | --- | --- | --- | --- | --- |
| 11 | Rogler | Harold | SW | 05-JAN-99 | (310) 456-7890 |
| 22 | Higgins | Heather | SE | 16-DEC-91 | (404) 524-8472 |
| 33 | Sullivan | Pat | NE | 21-FEB-88 | (305) 734-2987 |
| 44 | Speed | Kristen | MW | 14-JUN-90 | (708) 823-8222 |
| 55 | Sigafoos | Alex | NW | 05-MAR-01 | (310) 123-7890 |
| Primary key |  |  |  |  |  |
| Varchar2(2) | Varchar2(20), NOT NULL | Varchar2(20) | Char(2) | Date | Varchar2(14) |

# Structured Query Language, SQL

SQL, pronounced S.Q.L. or see' quel, is a language to access (retrieve) information in relational databases. SQL is also used to create and alter the structure of a table, and to insert, update, or delete data, and many other things. SQL is non-procedural: it contains no instructions on how to access and select the data. How to access and select the data is handled internally by SQL and is hidden from the user.

An SQL query for the INVENTORY table above is

**SELECT** DESCRIPTION, QUANTITY\_ON\_HAND

**FROM** INVENTORY

**WHERE** ITEM\_NUMBER = '10';

SQL is a **standard language** that exists as an American National Standards Institute (ANSI) standard and as an International Standards Organization (ISO) standard. Oracle and other developers of relational DBMS add enhancements to the standard SQL, and Oracle’s version is called **SQL\*PLUS**. In CS60, SQL is introduced in Chapter 07. With Microsoft Access, you can query a database either with its Query By Example graphical interface or by using SQL.

# Oracle's PL/SQL language

PL/SQL includes the selection structure in programming (e.g., IF-THEN-ELSE statement), repetition structure (loops), variables, and datatypes for records and arrays. User-defined functions, procedures, triggers (programs that run automatically when a database is changed), and other features are possible with PL/SQL. Some SQL statements can be embedded (included) in PL/SQL.

An example of **a selection statement** from PL/SQL is

A *condition that is* ***True*** *or* ***False***

The details of these statements are unimportant now. In CS60, PL/SQL is introduced in Chapter 09. Not all relational databases have such selection and repetition statements to manipulate the data.

**IF** quantity\_ordered <= 10**THEN**

price := 5.0;

#### **ELSE**

price := 4.0;

**END IF**;

An example of **a** **loop** from PL/SQL is

**FOR** i **IN** 1..100 **LOOP** sum := sum + i;**END LOOP**;

Visual Basic, C, COBOL, Java, and other languages also can be used to access data in databases.

# The parts of a database management system

A **database management system** (DBMS) is a program that allows you to

 set up and revise the structure of the database (including the number of tables and their names, number of columns, datatypes of the columns, names of the columns, constraints such as keys)

 add records to a table and delete records from a table. One way to do these is with a **form**, a display on a screen of one or more records or parts of those records. The data entry operator can edit the displayed values, create new rows, and enter data into them. Another way is to use SQL or PL/SQL.

 update (change) the values in the records.

 access existing data to respond to queries. A **query** is a question or instruction to retrieve data from tables. A query can be posed in SQL.

 access existing data to generate **reports**, output intended to be printed.

 manipulate the data to produce new data.

 provide masks to help the user enter phone numbers in the proper format.

 validate—test the data for its accuracy (does the data lie in a certain range, is the data an integer or some other data type),

 control whether an operator must supply entries for certain fields or whether a field can be left empty (does the data allow nulls?)

 trigger a change to one table as a result of a change to another (e.g., decrease the quantity on hand as a result of a sale). Programs called triggers are described in Chapter 10.

A diagram for a generic database management system (DBMS) is shown below.

**Forms** used to input, update, delete, and output data. Forms write SQL commands to accomplish their tasks and format.

**Reports** retrieve data by writing

SQL commands and formating the results

**The database file(s)**

**The user’s raw data, tables, and other objects such as forms, reports, and programs (procedures and functions) are stored in the database.**

**The database also includes many system tables, such as a table that stores the names of tables in the database**

By using SQL, **tables and other objects can be created and dropped, and raw data can be input, updated, deleted, and retrieved.**

**Graphical interfaces**

are SQL writing programs and formating programs.

They can create tables

and other objects.

**Natural Language Queries** interface to the database that accepts typed queries written in ordinary English, parses the sentence, and writes the SQL command for the query. It may format the results that are retrieved.

**Database Management System** (DBMS), the application program that manages the database.

If the DBMS is relational, it uses the language SQL,

and other languages may be part of the DBMS.

**Speech** **recognition** converts speech into text.

The speech is input into a microphone.

**Speech synthesis** converts text to speech. The speech is output from a loudspeaker or headphones.output from loudspeakers

Some raw data and programs accessed by the DBMS may be stored outside of the database in separate files.

Programs written in high-level languages can access the database essentially by writing SQL commands.

**Figure 1.1. A generic database management system with its data and metadata**

The data files created with many computer languages are composed of the raw data, and separate programs are used to store, modify, and view the data. However, a RDBMS integrates all raw data, table structures, index tables, data validation instructions, and instructions for forms, reports, and queries into one file or a few files.

# Steps to design a relational database

Many problems with databases are a result of their design—enough problems to warrant separate books on designing relational databases. In brief, the steps to design an RDBMS are:

1. Identify the information the user will get from the database. This identifies the fields of the table(s).

2. Organize the fields into one or more tables. Group the fields that are logically related into the same table. Eliminate or reduce redundancies by satisfying normalization rules that we will study in Chapter 04. Specify a datatype for each column (we will study data types in Chapter 06). Specify a primary key for each table.

3. If the database contains more than one table, establish relationships between the tables by specifying a foreign key that links to a primary key. This means to relate the tables or link the tables through common fields.

1. Create any necessary **indexes** for the table. An index allows the user to quickly search through the field that is indexed.

An index has two parts:

(1) the **index field** which has the values of the field or fields being indexed. An example of a useful index field on a big table of names is the Last name, First Name, Middle name.

(2) a **pointer field** with the record number or address for each record.

A table can have no fields indexed, one indexed, or more than one (if the table has more than one field). All fields could be indexed. In Oracle, an index is an object.

5. Define any data validation rules to ensure, as best as possible, that the data entered is accurate. Perhaps the data entered must be numeric in a certain range such as greater than 5 but less than 100, or with a certain format as for a phone number.

Tools to help design a relational database include:

(1) Paper and pencil, either freehand or using plastic templates, a ruler, and eraser.

(2) The drawing tools in Word, Powerpoint, and Visio Studio.

(3) Software programs that help design relational databases[[5]](#footnote-6)

# Terminal networks (optional)

**Mainframe Computer**

Printer(s), fax(es),

Scanner(s)

Modem(s)

Terminal n

Terminal 2

Terminal 1

n may be 100s or 1000s

Terminals are not monitors and terminal networks are not networks of computers. Early terminals with keyboards were dumb and often connected serially to a computer (through the serial port). Even the screen display was controlled by the computer, and 100’s of terminals were sometimes connected to one mainframe as a **terminal network**. All processing and file storage was on the mainframe.

Later the terminal acquired some intelligence and reduced the burden on the host. Still later, PC’s included terminal emulator programs that allowed them to serve as a terminal and access the shared data on the host. But also the PC’s could run their own applications. While networking the computers is a faster way to transfer data than a serial connection, sometimes the terminal network provides adequate transfer rates. Dr. Rogler set up a terminal network in a retail shop with a Pentium host running Unix, six other PC’s running DOS with a terminal emulator, three modems, and three printers. When a PC wanted to run Windows and applications, it could stop executing the emulator and begin local processing and local storage.

# File Server

A file server receives, stores, and sends files. At SMC, Zeus is a file server where students can upload files, and AcShare is a file server where students can download files.

# Client/Server Computing

**Client/server computing** reduces the amount of data transferred and better utilizes resources. The server stores the shared files. Either

 the server can accept input from the client, store and process the data, and transfer results to the client, or

 the processing can be shared by the client and server.

Internet

**Network Scanner, Fax, other peripherals**

**Network**

**Printer**

**Application Server**

**Web File Print executes programs**

**Server Server Server and**

**sends data to clients**

Terminator

**Client Computer 1 Client Computer 2 Client Computer n**

Clients usually can execute programs locally

**A bus network**

An **application server in a client/server system** executes programs and sends data to clients. At Santa Monica College, the database programs Oracle and Microsoft SQL Server are stored and execute on application servers, and the database itself (the data) is also stored on the server. A **file server** receives, stores, and sends files when a client requests a file. A **printer server** receives files to print, stores them on its harddrive, and feeds the data to a printer. The printer and server work together so the data does not overfill the printer’s buffer (memory). **Web servers** listen for messages from the Internet or intranet and send webpages. Santa Monica College has all these servers.

The server can be either dedicated to one task such as the database server or it can serve several applications. Many hundreds or thousands of clients can link to one server or a network of servers.

# Key Terms

backup

cluster

column

computer files

contiguous and noncontiguous clusters

cylinder

data compression

data item

data validation rules

database

database administrator (DBA)

database management system (DBMS)

database tables

datatype

defragmenting utility

delimiters between fields and records

designing a relational database

detailed and summary reports

directory of files

end-of-file (EOF) marker

field

file allocation table (FAT)

file closing

fixed-length and variable-length records

forms

fragmented and unfragmented files

index field

low-level formatting of the drive

magnetic secondary storage

mirrored drive arrays and striped drive arrays

nonvolatile, semipermanent, and volatile storage

offset

operating system

optical secondary storage

pointer field

primary index

primary memory and secondary storage

queries

random access and sequential access

random array of inexpensive drives (RAID)

read and write buffers

record

relational database

relational DBMS (RDBMS)

relational operators

report

row

secondary index

sector

table

[CS65 Online Dictionaries.doc](CS65%20Online%20Dictionaries.doc)

This MSWord file lists technical and general dictionaries, translators, and word pronouncers. For this hyperlink to work, the above file must be stored in the same folder or at the same hierarchical level as this CS60 Chapter 00.doc file.

1. The End-Of-File, Unused, and Bad entries are entered as codes [↑](#footnote-ref-2)
2. True client/server relational DBMS seriously consider how to back up and restore data. [↑](#footnote-ref-3)
3. Databases can have relationships between data, but a **knowledgebase** in an expert system or chatbot more intimately links data. In philosophy, epistemology is “the study or theory of the origin, nature, methods, and limits of knowledge.” **Cognitive science** combines the fields of computer science, philosophy, psychology, and neurology to study how knowledge is stored and accessed in biological brains, artificial neural networks, and conventional (Turing) computers. [↑](#footnote-ref-4)
4. One 8-bit byte can store the decimal numbers 0 to 255, or a total of 256 different numbers, each which can correspond to a different character as in the ASCII-8 character set or EBCDIC set or other collating sequence with 256 characters. Two bytes or 16 bits can store the decimal numbers 0 to 65,545 or 65,546 different numbers, each which can represent a different character. One form of Unicode uses two bytes to store each character. [↑](#footnote-ref-5)
5. Examples for Oracle are (1) Oracle's Designer by Oracle, which is the central topic in the course CS66, Advanced Oracle Programming at Santa Monica College, (2) ERwin by Logic Works, and (3) S-Designor by Sybase. [↑](#footnote-ref-6)