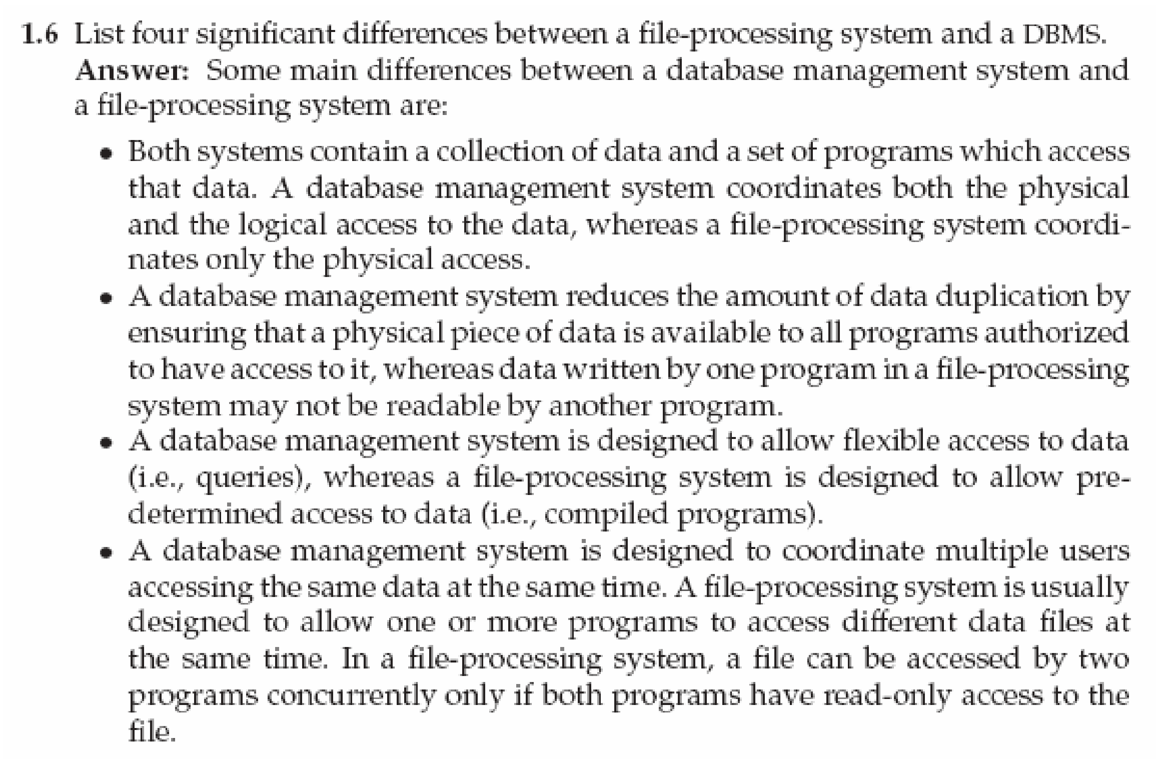

1.6 List four significant differences between a file-processing system and a DBMS.
Answer: Some main differences between a database management system and a file-processing system are:

- Both systems contain a collection of data and a set of programs which access that data. A database management system coordinates both the physical and the logical access to the data, whereas a file-processing system coordinates only the physical access.
- A database management system reduces the amount of data duplication by ensuring that a physical piece of data is available to all programs authorized to have access to it, whereas data written by one program in a file-processing system may not be readable by another program.
- A database management system is designed to allow flexible access to data (i.e., queries), whereas a file-processing system is designed to allow predetermined access to data (i.e., compiled programs).
- A database management system is designed to coordinate multiple users accessing the same data at the same time. A file-processing system is usually designed to allow one or more programs to access different data files at the same time. In a file-processing system, a file can be accessed by two programs concurrently only if both programs have read-only access to the file.

2.4 Describe the differences in meaning between the terms *relation* and *relation schema*.

Answer: A relation schema is a type definition, and a relation is an instance of that schema. For example, *student* (*ss#*, *name*) is a relation schema and

ss#	*name*
123-45-6789	Tom Jones
456-78-9123	Joe Brown

is a relation based on that schema.

2.5 Consider the relational database of Figure 2.35, where the primary keys are underlined. Give an expression in the relational algebra to express each of the following queries:

a. Find the names of all employees who work for First Bank Corporation.
b. Find the names and cities of residence of all employees who work for First Bank Corporation.
c. Find the names, street address, and cities of residence of all employees who work for First Bank Corporation and earn more than $10,000 per annum.
d. Find the names of all employees in this database who live in the same city as the company for which they work.
e. Assume the companies may be located in several cities. Find all companies located in every city in which Small Bank Corporation is located.

Answer:

a. $\Pi_{person\text{-}name}\ (\sigma_{company\text{-}name\ =\ \text{“First Bank Corporation”}}\ (works))$

employee (<u>*person-name*</u>, *street*, *city*)
works (<u>*person-name*</u>, *company-name*, *salary*)
company (<u>*company-name*</u>, *city*)
manages (<u>*person-name*</u>, *manager-name*)

Figure 2.35. Relational database for Exercises 2.1, 2.3 and 2.9.

b. $\Pi_{person\text{-}name,\ city}\ (employee\ \bowtie$
$(\sigma_{company\text{-}name\ =\ \text{“First Bank Corporation”}}\ (works)))$

c. $\Pi_{person\text{-}name,\ street,\ city}$
$(\sigma_{(company\text{-}name\ =\ \text{“First Bank Corporation”}\ \wedge\ salary\ >\ 10000)}$
$works \bowtie employee)$

d. $\Pi_{person\text{-}name}\ (employee \bowtie works \bowtie company)$

e. Note: Small Bank Corporation will be included in each answer.
$\Pi_{company\text{-}name}\ (company \div$
$(\Pi_{city}\ (\sigma_{company\text{-}name = \text{“Small Bank Corporation”}}\ (company))))$

2.6 Consider the relation of Figure 2.20, which shows the result of the query "Find the names of all customers who have a loan at the bank." Rewrite the query to include not only the name, but also the city of residence for each customer. Observe that now customer Jackson no longer appears in the result, even though Jackson does in fact have a loan from the bank.

a. Explain why Jackson does not appear in the result.
b. Suppose that you want Jackson to appear in the result. How would you modify the database to achieve this effect?
c. Again, suppose that you want Jackson to appear in the result. Write a query using an outer join that accomplishes this desire without your having to modify the database.

Answer: The rewritten query is
$\Pi_{customer\text{-}name, customer\text{-}city, amount}(borrower \bowtie loan \bowtie customer)$

a. Although Jackson does have a loan, no address is given for Jackson in the *customer* relation. Since no tuple in *customer* joins with the Jackson tuple of *borrower*, Jackson does not appear in the result.
b. The best solution is to insert Jackson's address into the *customer* relation. If the address is unknown, null values may be used. If the database system does not support nulls, a special value may be used (such as **unknown**) for Jackson's street and city. The special value chosen must not be a plausible name for an actual city or street.
c. $\Pi_{customer\text{-}name, customer\text{-}city, amount}((borrower \bowtie loan) \mathbin{⟕} customer)$

2.7 Consider the relational database of Figure 2.35. Give an expression in the relational algebra for each request:

a. Give all employees of First Bank Corporation a 10 percent salary raise.
b. Give all managers in this database a 10 percent salary raise, unless the salary would be greater than \$100,000. In such cases, give only a 3 percent raise.
c. Delete all tuples in the *works* relation for employees of Small Bank Corporation.

Answer:

a. $works \leftarrow \Pi_{person\text{-}name, company\text{-}name, 1.1*salary}($
$\quad \sigma_{(company\text{-}name = \text{"First Bank Corporation"})}(works))$
$\quad \cup\, (works - \sigma_{company\text{-}name = \text{"First Bank Corporation"}}(works))$

b. The same situation arises here. As before, t_1, holds the tuples to be updated and t_2 holds these tuples in their updated form.

$t_1 \leftarrow \Pi_{works.person\text{-}name, company\text{-}name, salary}$
$\quad (\sigma_{works.person\text{-}name = manager\text{-}name}(works \times manages))$
$t_2 \leftarrow \Pi_{works.person\text{-}name, company\text{-}name, salary*1.03}$
$\quad (\sigma_{t_1.salary\, *\, 1.1\, >\, 100000}(t_1))$
$t_2 \leftarrow t_2 \cup (\Pi_{works.person\text{-}name, company\text{-}name, salary*1.1}$
$\quad (\sigma_{t_1.salary\, *\, 1.1\, \leq\, 100000}(t_1)))$
$works \leftarrow (works - t_1) \cup t_2$

c. $works \leftarrow works - \sigma_{company-name = \text{"Small Bank Corporation"}}(works)$

2.8 Using the bank example, write relational-algebra queries to find the accounts held by more than two customers in the following ways:

a. Using an aggregate function.
b. Without using any aggregate functions.

Answer:

a. $t_1 \leftarrow {}_{account\text{-}number}\mathcal{G}_{\textbf{count}\ customer\text{-}name}(depositor)$
$\Pi_{account\text{-}number}\left(\sigma_{num\text{-}holders>2}\left(\rho_{account\text{-}holders(account\text{-}number,num\text{-}holders)}(t_1)\right)\right)$

b. $t_1 \leftarrow (\rho_{d1}(depositor) \times \rho_{d2}(depositor) \times \rho_{d3}(depositor))$
$t_2 \leftarrow \sigma_{(d1.account\text{-}number=d2.account\text{-}number=d3.account\text{-}number)}(t_1)$
$\Pi_{d1.account\text{-}number}(\sigma_{(d1.customer\text{-}name\neq d2.customer\text{-}name\ \wedge}$
$_{d2.customer\text{-}name\neq d3.customer\text{-}name\ \wedge d3.customer\text{-}name\neq d1.customer\text{-}name)}(t_2))$

2.9 Consider the relational database of Figure 2.35. Give a relational-algebra expression for each of the following queries:

a. Find the company with the most employees.
b. Find the company with the smallest payroll.
c. Find those companies whose employees earn a higher salary, on average, than the average salary at First Bank Corporation.

Answer:

a. $t_1 \leftarrow {}_{company\text{-}name}\mathcal{G}_{\textbf{count-distinct}\ person\text{-}name}(works)$
$t_2 \leftarrow \textbf{max}_{num\text{-}employees}(\rho_{company\text{-}strength(company\text{-}name,num\text{-}employees)}(t_1))$
$\Pi_{company\text{-}name}(\rho_{t_3(company\text{-}name,num\text{-}employees)}(t_1) \bowtie \rho_{t_4(num\text{-}employees)}(t_2))$

b. $t_1 \leftarrow {}_{company\text{-}name}\mathcal{G}_{\textbf{sum}\ salary}(works)$
$t_2 \leftarrow \textbf{min}_{payroll}(\rho_{company\text{-}payroll(company\text{-}name,payroll)}(t_1))$
$\Pi_{company\text{-}name}(\rho_{t_3(company\text{-}name,payroll)}(t_1) \bowtie \rho_{t_4(payroll)}(t_2))$

c. $t_1 \leftarrow {}_{company\text{-}name}\mathcal{G}_{\textbf{avg}\ salary}(works)$
$t_2 \leftarrow \sigma_{company\text{-}name\ =\ \text{“First Bank Corporation”}}(t_1)$
$\Pi_{t_3.company\text{-}name}((\rho_{t_3(company\text{-}name,avg\text{-}salary)}(t_1))$
$\bowtie_{t_3.avg\text{-}salary\ >\ first\text{-}bank.avg\text{-}salary}\ (\rho_{first\text{-}bank(company\text{-}name,avg\text{-}salary)}(t_2)))$

2.10 List two reasons why null values might be introduced into the database.
Answer: Nulls may be introduced into the database because the actual value is either unknown or does not exist. For example, an employee whose address has changed and whose new address is not yet known should be retained with a null address. If employee tuples have a composite attribute *dependents*, and a particular employee has no dependents, then that tuple's *dependents* attribute should be given a null value.