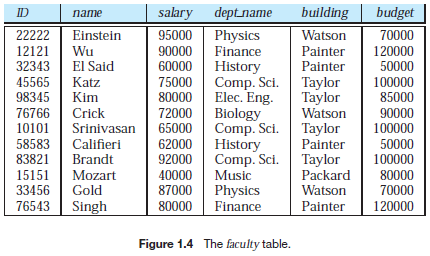
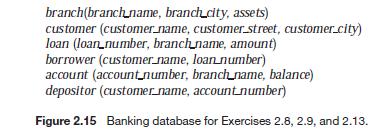
1. **1.10** List five responsibilities of a database-management system. For each responsibility, explain the problems that would arise if the responsibility were not discharged.

* Store and retrieve data (conveniently and efficiently)
  + If this were to not be completed, then the entire system would not work. If data were stored in inconvenient file systems of old or divergent systems (data isolation), then all other responsibilities break down, including the ability to easily retrieve data, update, or search.
* Maintain data integrity
  + Without the ability to backup for system crashes and unauthorized access, data can be corrupted. Consistency constraints allow a developer to enforce standards in data maintenance. Atomicity is also an important part allowing transactions within the data to occur only if certain conditions are met – without which the integrity fails.
* Security and access control
  + DBSM should provide access of varying levels (DB managers, analysts, naïve users). Without this, anyone could access, alter or corrupt data at any level possibly create confidentiality leaks.
* Concurrent Access
  + Without the ability for multiple users to access, data anomalies occur, providing data integrity issues.
* Centralized Data
  + If the physical level of a database is not centralized and concise, then data can be stored in multiple places and creates issues in maintaining and updating databases as well as integrity issues that may arise from each transaction or query.

1. **1.12** Explain what problems are caused by the design of the table in Figure 1.4.

* Budgets appear to belong to each individual ID or named faculty.
* Information is repeated, taking up more space and making it more difficult to update all entries (ie, dept\_name or building). If any professor teaches for more than one department, primary key issues.
* Added data may not have all these fields and would require null values, which complicate queries.

1. **2.9** Consider the bank database of Figure 2.15. 

a. What are the appropriate primary keys?

Recall a primary key is a candidate key chosen as the principle method of identifying tuples.

|  |  |
| --- | --- |
| **Relation** | **Primary Key** |
| branch | branch\_name |
| customer | customer\_name, customer\_street |
| loan | loan\_number |
| borrower  (assuming there may be more than one borrower on any given loan) | customer\_name, loan\_number |
| depositor | customer\_name, account\_number |

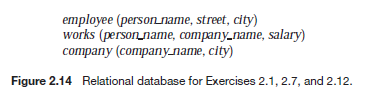
b. Given your choice of primary keys, identify appropriate foreign keys.

Recall a foreign key is the primary key from another relation:

customer\_name, loan\_number, account\_number,

1. **2.10** Consider the *advisor* relation shown in Figure 2.8, with *s id* as the primary key of *advisor*. Suppose a student can have more than one advisor. Then, would *s\_id* still be a primary key of the *advisor* relation? If not, what should the primary key of *advisor* be?

No, s\_id would no longer be a good primary key by itself. A primary key must be able to uniquely identify each tuple in the relation. You would however, be able to use s\_id and i\_id or add an additional field, (like major, or dept for those of us with an advisor in both fields) in order to uniquely identify each tuple.

1. Draw a schema diagram for the database in Figure 2.14, page 53 of the textbook. 

You can assume that values of person\_name in the relation employee are unique.

