

generate their own unique identifiers. An alternative is to use some unique combination of other attributes as a key.

The primary key should be chosen such that its attribute values are never, or are very rarely, changed. For instance, the address field of a person should not be part of the primary key, since it is likely to change. Social security numbers, on the other hand, are guaranteed never to change. Unique identifiers generated by enterprises generally do not change, except if two enterprises merge; in such a case the same identifier may have been issued by both enterprises, and a reallocation of identifiers may be required to make sure they are unique.

Figure 2.8 shows the complete set of relations that we use in our sample university schema, with primary-key attributes underlined.

Next, we consider another type of constraint on the contents of relations, called foreign-key constraints. Consider the attribute *dept\_name* of the *instructor* relation. It would not make sense for a tuple in *instructor* to have a value for *dept\_name* that does not correspond to a department in the *department* relation. Thus, in any database instance, given any tuple, say  $t_a$ , from the *instructor* relation, there must be some tuple, say  $t_b$ , in the *department* relation such that the value of the *dept\_name* attribute of  $t_a$  is the same as the value of the primary key, *dept\_name*, of  $t_b$ .

A **foreign-key constraint** from attribute(s)  $A$  of relation  $r_1$  to the primary-key  $B$  of relation  $r_2$  states that on any database instance, the value of  $A$  for each tuple in  $r_1$  must also be the value of  $B$  for some tuple in  $r_2$ . Attribute set  $A$  is called a **foreign key** from  $r_1$ , referencing  $r_2$ . The relation  $r_1$  is also called the **referencing relation** of the foreign-key constraint, and  $r_2$  is called the **referenced relation**.

For example, the attribute *dept\_name* in *instructor* is a foreign key from *instructor*, referencing *department*; note that *dept\_name* is the primary key of *department*. Similarly,

---

```

classroom(building, room_number, capacity)
department(dept_name, building, budget)
course(course_id, title, dept_name, credits)
instructor(ID, name, dept_name, salary)
section(course_id, sec_id, semester, year, building, room_number, time_slot_id)
teaches(ID, course_id, sec_id, semester, year)
student(ID, name, dept_name, tot_cred)
takes(ID, course_id, sec_id, semester, year, grade)
advisor(s_ID, i_ID)
time_slot(time_slot_id, day, start_time, end_time)
prereq(course_id, prereq_id)

```

---

**Figure 2.8** Schema of the university database.