APPENDIX A

Detailed University Schema

In this appendix, we present the full details of our running-example university database. In Section A.1 we present the full schema as used in the text and the E-R diagram that corresponds to that schema. In Section A.2 we present a relatively complete SQL data definition for our running university example. Besides listing a datatype for each attribute, we include a substantial number of constraints. Finally, in Section A.3 we present sample data that correspond to our schema. SQL scripts to create all the relations in the schema, and to populate them with sample data, are available on the Web site of the book, db-book.com.

A.1 Full Schema

The full schema of the University database as used in the text is shown in Figure A.1. The E-R diagram that corresponds to that schema, and used throughout the text, is shown in Figure A.2.

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 A.1 Schema of the University database.

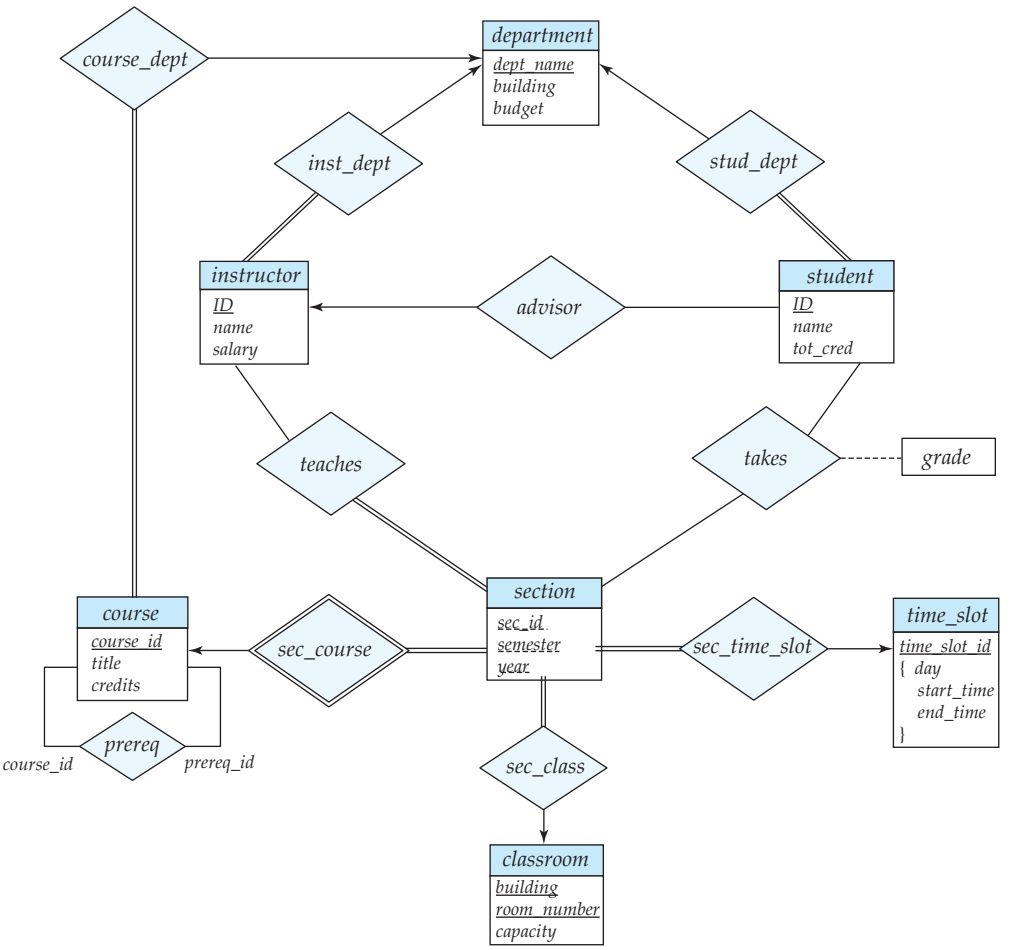

Figure A.2 E-R diagram for a university enterprise.

A.2 DDL

In this section, we present a relatively complete SQL data definition for our example. Besides listing a datatype for each attribute, we include a substantial number of constraints.

```
create table classroom
    (building       varchar (15),
     room_number    varchar (7),
     capacity       numeric (4,0),
     primary key (building, room_number));
```

```
create table department
    (dept_name      varchar (20),
     building       varchar (15),
     budget         numeric (12,2) check (budget > 0),
     primary key (dept_name));

create table course
    (course_id      varchar (8),
     title          varchar (50),
     dept_name      varchar (20),
     credits        numeric (2,0) check (credits > 0),
     primary key (course_id),
     foreign key (dept_name) references department
                    on delete set null);

create table instructor
    (ID             varchar (5),
     name           varchar (20) not null,
     dept_name      varchar (20),
     salary         numeric (8,2) check (salary > 29000),
     primary key (ID),
     foreign key (dept_name) references department
                    on delete set null);

create table section
    (course_id      varchar (8),
     sec_id         varchar (8),
     semester       varchar (6) check (semester in
                            ('Fall', 'Winter', 'Spring', 'Summer')),
     year           numeric (4,0) check (year > 1701 and year < 2100),
     building       varchar (15),
     room_number    varchar (7),
     time_slot_id   varchar (4),
     primary key (course_id, sec_id, semester, year),
     foreign key (course_id) references course
                    on delete cascade,
     foreign key (building, room_number) references classroom
                    on delete set null);
```

In the above DDL we add the **on delete cascade** specification to a foreign key constraint if the existence of the tuple depends on the referenced tuple. For example we add the **on delete cascade** specification to the foreign key constraint from *section* (which was generated from weak entity *section*), to *course* (which was

its identifying relationship). In other foreign key constraints we either specify **on delete set null**, which allows deletion of a referenced tuple by setting the referencing value to null, or do not add any specification, which prevents the deletion of any referenced tuple. For example, if a department is deleted, we would not wish to delete associated instructors; the foreign key constraint from *instructor* to *department* instead sets the *dept_name* attribute to null. On the other hand, the foreign key constraint for the *prereq* relation, shown later, prevents the deletion of a course that is required as a prerequisite for another course. For the *advisor* relation, shown later, we allow *i_ID* to be set to null if an instructor is deleted, but delete an *advisor* tuple if the referenced student is deleted.

```
create table teaches
    (ID             varchar (5),
     course_id      varchar (8),
     sec_id         varchar (8),
     semester       varchar (6),
     year           numeric (4,0),
     primary key (ID, course_id, sec_id, semester, year),
     foreign key (course_id, sec_id, semester, year) references section
                    on delete cascade,
     foreign key (ID) references instructor
                    on delete cascade);

create table student
    (ID             varchar (5),
     name           varchar (20) not null,
     dept_name      varchar (20),
     tot_cred       numeric (3,0) check (tot_cred >= 0),
     primary key (ID),
     foreign key (dept_name) references department
                    on delete set null);

create table takes
    (ID             varchar (5),
     course_id      varchar (8),
     sec_id         varchar (8),
     semester       varchar (6),
     year           numeric (4,0),
     grade          varchar (2),
     primary key (ID, course_id, sec_id, semester, year),
     foreign key (course_id, sec_id, semester, year) references section
                    on delete cascade,
     foreign key (ID) references student
                    on delete cascade);
```

```
create table advisor
    (s_ID          varchar (5),
     i_ID          varchar (5),
     primary key (s_ID),
     foreign key (i_ID) references instructor (ID)
                   on delete set null,
     foreign key (s_ID) references student (ID)
                   on delete cascade);

create table prereq
    (course_id     varchar(8),
     prereq_id     varchar(8),
     primary key (course_id, prereq_id),
     foreign key (course_id) references course
                   on delete cascade,
     foreign key (prereq_id) references course);
```

The following **create table** statement for the table *time_slot* can be run on most database systems, but does not work on Oracle (at least as of Oracle version 11), since Oracle does not support the SQL standard type **time**.

```
create table timeslot
    (time_slot_id  varchar (4),
     day           varchar (1) check (day in ('M', 'T', 'W', 'R', 'F', 'S', 'U')),
     start_time    time,
     end_time      time,
     primary key (time_slot_id, day, start_time));
```

The syntax for specifying time in SQL is illustrated by these examples: '08:30', '13:55', and '5:30 PM'. Since Oracle does not support the **time** type, for Oracle we use the following schema instead:

```
create table timeslot
    (time_slot_id  varchar (4),
     day           varchar (1),
     start_hr      numeric (2) check (start_hr >= 0 and end_hr < 24),
     start_min     numeric (2) check (start_min >= 0 and start_min < 60),
     end_hr        numeric (2) check (end_hr >= 0 and end_hr < 24),
     end_min       numeric (2) check (end_min >= 0 and end_min < 60),
     primary key (time_slot_id, day, start_hr, start_min));
```

The difference is that *start_time* has been replaced by two attributes *start_hr* and *start_min*, and similarly *end_time* has been replaced by attributes *end_hr* and *end_min*. These attributes also have constraints that ensure that only numbers representing valid time values appear in those attributes. This version of the

schema for *time_slot* works on all databases, including Oracle. Note that although Oracle supports the **datetime** datatype, **datetime** includes a specific day, month, and year as well as a time, and is not appropriate here since we want only a time. There are two alternatives to splitting the time attributes into an hour and a minute component, but neither is desirable. The first alternative is to use a **varchar** type, but that makes it hard to enforce validity constraints on the string as well as to perform comparison on time. The second alternative is to encode time as an integer representing a number of minutes (or seconds) from midnight, but this alternative requires extra code with each query to covert values between the standard time representation and the integer encoding. We therefore chose the two-part solution.

A.3 Sample Data

In this section we provide sample data for each of the relations defined in the previous section.

building	*room_number*	*capacity*
Packard	101	500
Painter	514	10
Taylor	3128	70
Watson	100	30
Watson	120	50

Figure A.3 The *classroom* relation.

dept_name	*building*	*budget*
Biology	Watson	90000
Comp. Sci.	Taylor	100000
Elec. Eng.	Taylor	85000
Finance	Painter	120000
History	Painter	50000
Music	Packard	80000
Physics	Watson	70000

Figure A.4 The *department* relation.

course_id	*title*	*dept_name*	*credits*
BIO-101	Intro. to Biology	Biology	4
BIO-301	Genetics	Biology	4
BIO-399	Computational Biology	Biology	3
CS-101	Intro. to Computer Science	Comp. Sci.	4
CS-190	Game Design	Comp. Sci.	4
CS-315	Robotics	Comp. Sci.	3
CS-319	Image Processing	Comp. Sci.	3
CS-347	Database System Concepts	Comp. Sci.	3
EE-181	Intro. to Digital Systems	Elec. Eng.	3
FIN-201	Investment Banking	Finance	3
HIS-351	World History	History	3
MU-199	Music Video Production	Music	3
PHY-101	Physical Principles	Physics	4

Figure A.5 The *course* relation.

ID	*name*	*dept_name*	*salary*
10101	Srinivasan	Comp. Sci.	65000
12121	Wu	Finance	90000
15151	Mozart	Music	40000
22222	Einstein	Physics	95000
32343	El Said	History	60000
33456	Gold	Physics	87000
45565	Katz	Comp. Sci.	75000
58583	Califieri	History	62000
76543	Singh	Finance	80000
76766	Crick	Biology	72000
83821	Brandt	Comp. Sci.	92000
98345	Kim	Elec. Eng.	80000

Figure A.6 The *instructor* relation.

course_id	*sec_id*	*semester*	*year*	*building*	*room_number*	*time_slot_id*
BIO-101	1	Summer	2009	Painter	514	B
BIO-301	1	Summer	2010	Painter	514	A
CS-101	1	Fall	2009	Packard	101	H
CS-101	1	Spring	2010	Packard	101	F
CS-190	1	Spring	2009	Taylor	3128	E
CS-190	2	Spring	2009	Taylor	3128	A
CS-315	1	Spring	2010	Watson	120	D
CS-319	1	Spring	2010	Watson	100	B
CS-319	2	Spring	2010	Taylor	3128	C
CS-347	1	Fall	2009	Taylor	3128	A
EE-181	1	Spring	2009	Taylor	3128	C
FIN-201	1	Spring	2010	Packard	101	B
HIS-351	1	Spring	2010	Painter	514	C
MU-199	1	Spring	2010	Packard	101	D
PHY-101	1	Fall	2009	Watson	100	A

Figure A.7 The *section* relation.

ID	*course_id*	*sec_id*	*semester*	*year*
10101	CS-101	1	Fall	2009
10101	CS-315	1	Spring	2010
10101	CS-347	1	Fall	2009
12121	FIN-201	1	Spring	2010
15151	MU-199	1	Spring	2010
22222	PHY-101	1	Fall	2009
32343	HIS-351	1	Spring	2010
45565	CS-101	1	Spring	2010
45565	CS-319	1	Spring	2010
76766	BIO-101	1	Summer	2009
76766	BIO-301	1	Summer	2010
83821	CS-190	1	Spring	2009
83821	CS-190	2	Spring	2009
83821	CS-319	2	Spring	2010
98345	EE-181	1	Spring	2009

Figure A.8 The *teaches* relation.

ID	*name*	*dept_name*	*tot_cred*
00128	Zhang	Comp. Sci.	102
12345	Shankar	Comp. Sci.	32
19991	Brandt	History	80
23121	Chavez	Finance	110
44553	Peltier	Physics	56
45678	Levy	Physics	46
54321	Williams	Comp. Sci.	54
55739	Sanchez	Music	38
70557	Snow	Physics	0
76543	Brown	Comp. Sci.	58
76653	Aoi	Elec. Eng.	60
98765	Bourikas	Elec. Eng.	98
98988	Tanaka	Biology	120

Figure A.9 The *student* relation.

ID	*course_id*	*sec_id*	*semester*	*year*	*grade*
00128	CS-101	1	Fall	2009	A
00128	CS-347	1	Fall	2009	A-
12345	CS-101	1	Fall	2009	C
12345	CS-190	2	Spring	2009	A
12345	CS-315	1	Spring	2010	A
12345	CS-347	1	Fall	2009	A
19991	HIS-351	1	Spring	2010	B
23121	FIN-201	1	Spring	2010	C+
44553	PHY-101	1	Fall	2009	B-
45678	CS-101	1	Fall	2009	F
45678	CS-101	1	Spring	2010	B+
45678	CS-319	1	Spring	2010	B
54321	CS-101	1	Fall	2009	A-
54321	CS-190	2	Spring	2009	B+
55739	MU-199	1	Spring	2010	A-
76543	CS-101	1	Fall	2009	A
76543	CS-319	2	Spring	2010	A
76653	EE-181	1	Spring	2009	C
98765	CS-101	1	Fall	2009	C-
98765	CS-315	1	Spring	2010	B
98988	BIO-101	1	Summer	2009	A
98988	BIO-301	1	Summer	2010	*null*

Figure A.10 The *takes* relation.

s_id	*i_id*
00128	45565
12345	10101
23121	76543
44553	22222
45678	22222
76543	45565
76653	98345
98765	98345
98988	76766

Figure A.11 The *advisor* relation.

time_slot_id	*day*	*start_time*	*end_time*
A	M	8:00	8:50
A	W	8:00	8:50
A	F	8:00	8:50
B	M	9:00	9:50
B	W	9:00	9:50
B	F	9:00	9:50
C	M	11:00	11:50
C	W	11:00	11:50
C	F	11:00	11:50
D	M	13:00	13:50
D	W	13:00	13:50
D	F	13:00	13:50
E	T	10:30	11:45
E	R	10:30	11:45
F	T	14:30	15:45
F	R	14:30	15:45
G	M	16:00	16:50
G	W	16:00	16:50
G	F	16:00	16:50
H	W	10:00	12:30

Figure A.12 The *time_slot* relation.

course_id	*prereq_id*
BIO-301	BIO-101
BIO-399	BIO-101
CS-190	CS-101
CS-315	CS-101
CS-319	CS-101
CS-347	CS-101
EE-181	PHY-101

Figure A.13 The *prereq* relation.

time_slot_id	*day*	*start_hr*	*start_min*	*end_hr*	*end_min*
A	M	8	0	8	50
A	W	8	0	8	50
A	F	8	0	8	50
B	M	9	0	9	50
B	W	9	0	9	50
B	F	9	0	9	50
C	M	11	0	11	50
C	W	11	0	11	50
C	F	11	0	11	50
D	M	13	0	13	50
D	W	13	0	13	50
D	F	13	0	13	50
E	T	10	30	11	45
E	R	10	30	11	45
F	T	14	30	15	45
F	R	14	30	15	45
G	M	16	0	16	50
G	W	16	0	16	50
G	F	16	0	16	50
H	W	10	0	12	30

Figure A.14 The *time_slot* relation with start and end time separated into hour and minute.