1. Write the following queries in SQL, using the university schema.

a. Find the titles of courses in the Comp. Sci. department that have 3 credits.

b. Find the IDs of all students who were taught by an instructor named Einstein;make sure there are no duplicates in the result.

c. Find the highest salary of any instructor.

d. Find all instructors earning the highest salary (there may be more than one with the same salary).

e. Find the enrollment of each section that was offered in Autumn 2009.

f. Find the maximum enrollment, across all sections, in Autumn 2009.

g. Find the sections that had themaximum enrollment in Autumn 2009.

SQL data definition for part of the university database.

**create table** *department*

(*dept name* **varchar** (20),

*building* **varchar** (15),

*budget* **numeric** (12,2),

**primary key** (*dept name*));

**create table** *course*

(*course id* **varchar** (7),

*title* **varchar** (50),

*dept name* **varchar** (20),

*credits* **numeric** (2,0),

**primary key** (*course id*),

**foreign key** (*dept name*) **references** *department*);

**create table** *instructor*

(*ID* **varchar** (5),

*name* **varchar** (20) **not null**,

*dept name* **varchar** (20),

*salary* **numeric** (8,2),

**primary key** (*ID*),

**foreign key** (*dept name*) **references** *department*);

**create table** *section*

(*course id* **varchar** (8),

*sec id* **varchar** (8),

*semester* **varchar** (6),

*year* **numeric** (4,0),

*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*);

**create table** *teaches*

(*ID* **varchar** (5),

*course id* **varchar** (8),

*sec id* **varchar** (8),

*semester* **varchar** (6),

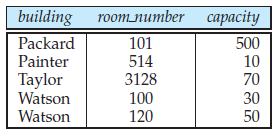
*year* **numeric** (4,0),

**primary key** (*ID*, *course id*, *sec id*, *semester*, *year*),

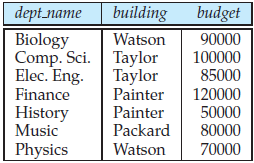
**foreign key** (*course id*, *sec id*, *semester*, *year*) **references** *section*,

**foreign key** (*ID*) **references** *instructor*);

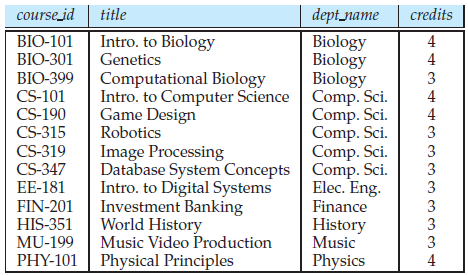
**The *classroom* relation.**

****

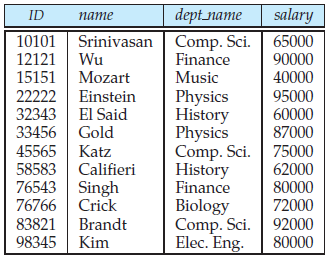
**The *department* relation.**

****

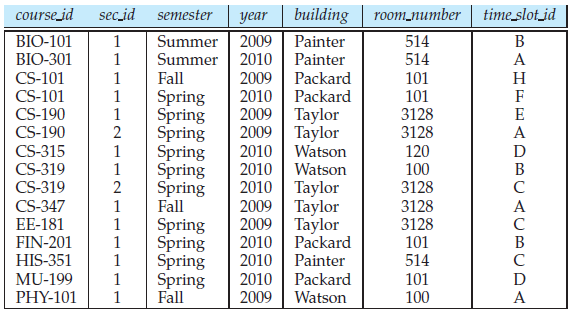
**The *course* relation.**

****

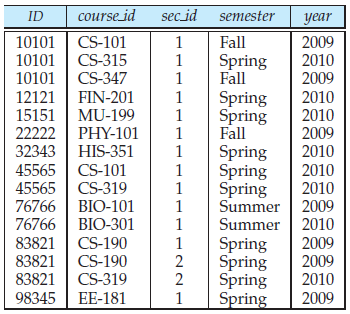
**The *instructor* relation.**

****

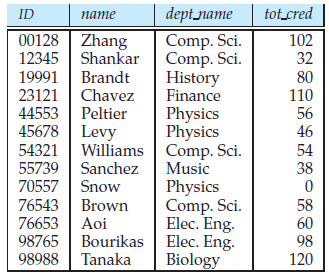
**The *section* relation.**

****

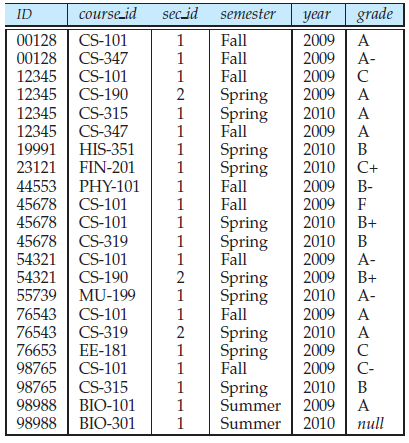
**The *teaches* relation.**

****

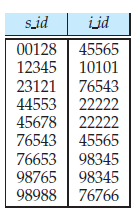
**The *student* relation.**

****

**The *takes* relation.**

****

**The *advisor* relation.**

****

**Answer:**

a. Find the titles of courses in the Comp. Sci. department that have 3 credits.

**select** *title*

**from** *course*

**where** *dept name* = ’Comp. Sci.’**and** *credits* = 3

b. Find the IDs of all students who were taught by an instructor named Einstein; make sure there are no duplicates in the result. This query can be answered in several differentways. One way is as follows.

**select distinct** *student.ID*

**from** (*student* **join** *takes* **using**(*ID*))

**join** (*instructor* **join** *teaches* **using**(*ID*))

**using**(*course id*, *sec id*, *semester*, *year*)

**where** *instructor.name* = ’Einstein’

As an alternative to th **join .. using** syntax above the query can be written by enumerating relations in the **from**clause, and adding the corresponding join predicates on *ID*, *course id*, *section id*, *semester*, and

*year* to the **where** clause.

Note that using natural join in place of **join .. using** would result in equating student *ID* with instructor *ID*, which is incorrect.

c. Find the highest salary of any instructor.

**select max**(*salary*) **from** *instructor*

d. Find all instructors earning the highest salary (there may be more than one with the same salary).

**select** *ID*, *name*

**from** *instructor*

**where** *salary* = (**select max**(*salary*) **from** *instructor*)

e. Find the enrollment of each section thatwas offered inAutumn 2009. One way of writing the query is as follows.

**select** *course id*, *sec id*, **count**(*ID*)

**from** *section* **natural join** *takes*

**where** *semester* = ’Autumn’

**and** *year* = 2009

**group by** *course id*, *sec id*

f. Find the maximum enrollment, across all sections, in Autumn 2009. One way of writing this query is as follows:

**select max**(*enrollment*)

**from** (**select count**(*ID*) **as** *enrollment*

**from** *section* **natural join** *takes*

**where** *semester* = ’Autumn’

**and** *year* = 2009

**group by** *course id*, *sec id*)

g. Find the sections that had themaximumenrollment inAutumn 2009. The following answer uses a **with** clause to create a temporary view, simplifying the query.

**with** *sec enrollment* **as** (

**select** *course id*, *sec id*, **count**(*ID*) **as** *enrollment*

**from** *section* **natural join** *takes*

**where** *semester* = ’Autumn’

**and** *year* = 2009

**group by** *course id*, *sec id*)

**select** *course id*, *sec id*

**from** *sec enrollment*

**where** *enrollment* = (**select max**(*enrollment*) **from** *sec enrollment*)