Introduction to SQL

SQL

- Overview of the SQL Query Language
- Data Definition
- Basic Query Structure
- Additional Basic Operations
- Set Operations
- Null Values
- Aggregate Functions
- Nested Subqueries
- Modification of the Database
- Advanced SQL : many useful features

Data Definition Language

The SQL data-definition language (DDL) allows defining :

- The schema for each relation.
- The domain of values associated with each attribute.
- Integrity constraints
- also other information such as
  - The set of indices to be maintained for each relations.
  - Security and authorization information for each relation.
  - The physical storage structure of each relation on disk.

Domain Types in SQL

- char(n). Fixed length character string, with user-specified length *n.*
- varchar(n). Variable length character strings, with user-specified maximum length *n.*
- int. Integer (a finite subset of the integers that is machine-dependent).
- numeric(p,d). Fixed point number, with user-specified precision of *p* digits, with *d* digits to the right of decimal point.
- float(n). Floating point number, with user-specified precision of at least *n* digits.

Create Table statement

- Define tables, their columns, keys, etc

- Example:

```
create table instructor (
        ID              char(5),
        name            varchar(20),
        dept_name       varchar(20),
        salary          numeric(8,2))
```

SQL names are case insensitive

Exercise : define student and dept tables from previous module
(be aware of similarly named columns)

Integrity Constraints in Create Table

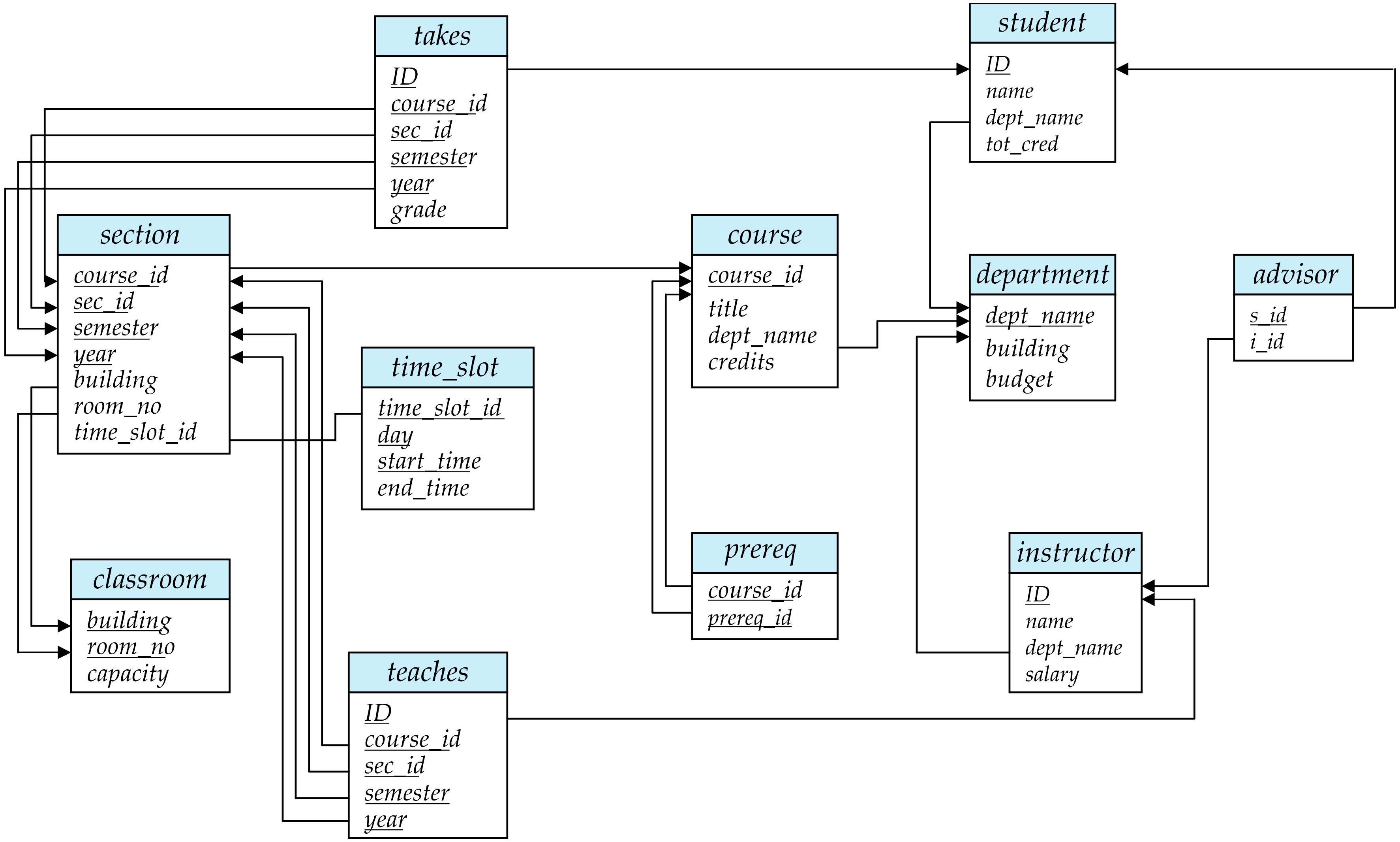
- not null
- primary key (A_1, ..., A_n)
- foreign key (A_m, ..., A_n) references *r*

```
create table instructor (
        ID              char(5),
        name            varchar(20) not null,
        dept_name       varchar(20),
        salary          numeric(8,2),
        primary key (ID),
        foreign key (dept_name) references department)
```

- We declare *dept_name* as the primary key for *department*

- primary key declaration on an attribute automatically ensures not null

Schema Diagram for University Database



takes
ID
course_id
sec_id
semester
year
grade

student
ID
name
dept_name
tot_cred

section
course_id
sec_id
semester
year
building
room_no
time_slot_id

course
course_id
title
dept_name
credits

department
dept_name
building
budget

advisor
s_id
i_id

time_slot
time_slot_id
day
start_time
end_time

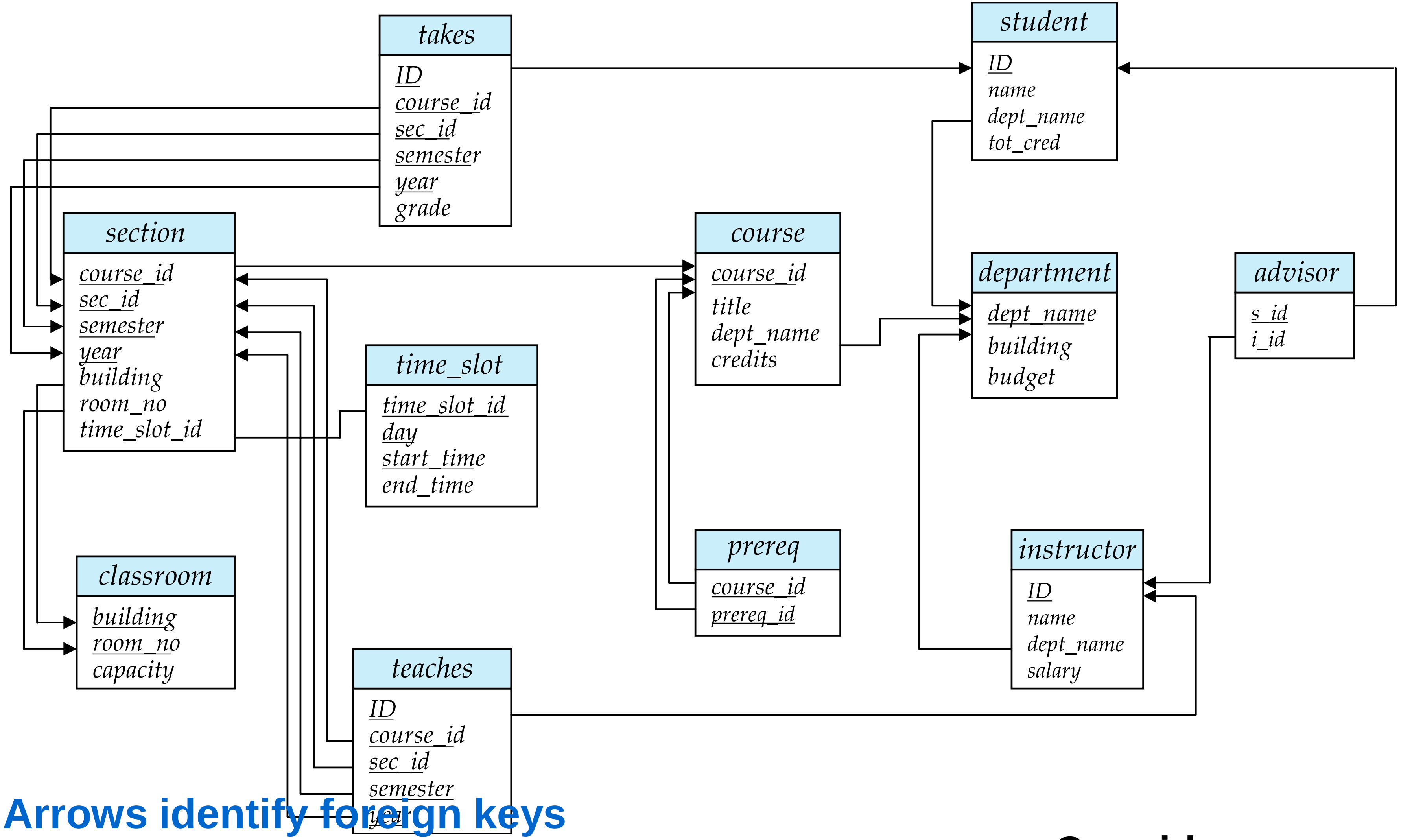
prereq
course_id
prereq_id

instructor
ID
name
dept_name
salary

classroom
building
room_no
capacity

teaches
ID
course_id
sec_id
semester
year

Schema Diagram for University Database



takes
- ID
- course_id
- sec_id
- semester
- year
- grade

student
- ID
- name
- dept_name
- tot_cred

section
- course_id
- sec_id
- semester
- year
- building
- room_no
- time_slot_id

course
- course_id
- title
- dept_name
- credits

department
- dept_name
- building
- budget

advisor
- s_id
- i_id

time_slot
- time_slot_id
- day
- start_time
- end_time

classroom
- building
- room_no
- capacity

prereq
- course_id
- prereq_id

instructor
- ID
- name
- dept_name
- salary

teaches
- ID
- course_id
- sec_id
- semester
- year

- Arrows identify foreign keys

- ensure that a std cannot take two sections of the same course !

Consider some example data

Data Insertion

- Add a new *course*

```
insert into course
    values (’CS-437’, ’DB Systems’, ’Comp. Sci.’, 4);
```

- or equivalently

```
insert into course
          (course_id, title, dept_name, credits)
 values (’CS-437’, ’DB Systems’, ’Comp. Sci.’, 4);
```

- Add a new *student* with *tot_creds* set to null

```
insert into student
    values (’3003’, ’Green’, ’Finance’, null);
```

Drop and Alter Table Constructs

- drop table *student*
  - Deletes the table and its contents
- delete from *student*
  - Deletes all contents of table, but retains table definition
- alter table
  - alter table *r* add *A D*
    - add attribute A of type D.
    - New attribute given null values.
  - alter table *r* drop *A*