Lecture 12

The Structured Query Language (SQL)

Week 6

SQL: Structured Query Language

- Both a Data Definition Language (DDL) and Data Manipulation Language (DML)
- We study SQL2: Standard:
  - ANSI X3,135-1992 (American / US)
  - ISO 9075:1992 (International)

Data Definition in SQL

- Defines a schema and the tables belonging to it

Some Remarks

- Relation is called a Table
- Attribute is called a column
- Tuple is called a ***row***
  (we use these interchangeably)
- A Table may contain identical rows
  (i.e. not a set of tuples, but a *bag* of tuples)
- We can declare if we want unique tuples

Catalog

- A Catalog is the data dictionary describing all databases maintained by-, or known to a database management system
- SQL2: Each database has a ***Schema***

CREATE SCHEMA company AUTHORIZATION smith

Schema name

Owner of schema

- Oracle SQL*Plus: one schema per user

CREATE user smith identified by <passwd>

User *and schema* name

Schema

- A Schema consists of tables and associated constraints

Table of schema ‘company’ (SQL2)

CREATE TABLE company.employee …

or

CREATE TABLE employee

Table of currently ‘open’ schema
(and Oracle SQL*Plus)

Attributes (columns)

- Attributes (columns) have a data type

  CHAR, CHAR(n), VARCHAR(n), INT, BIT(n), DATE, TIME, FLOAT, DOUBLE, etc…

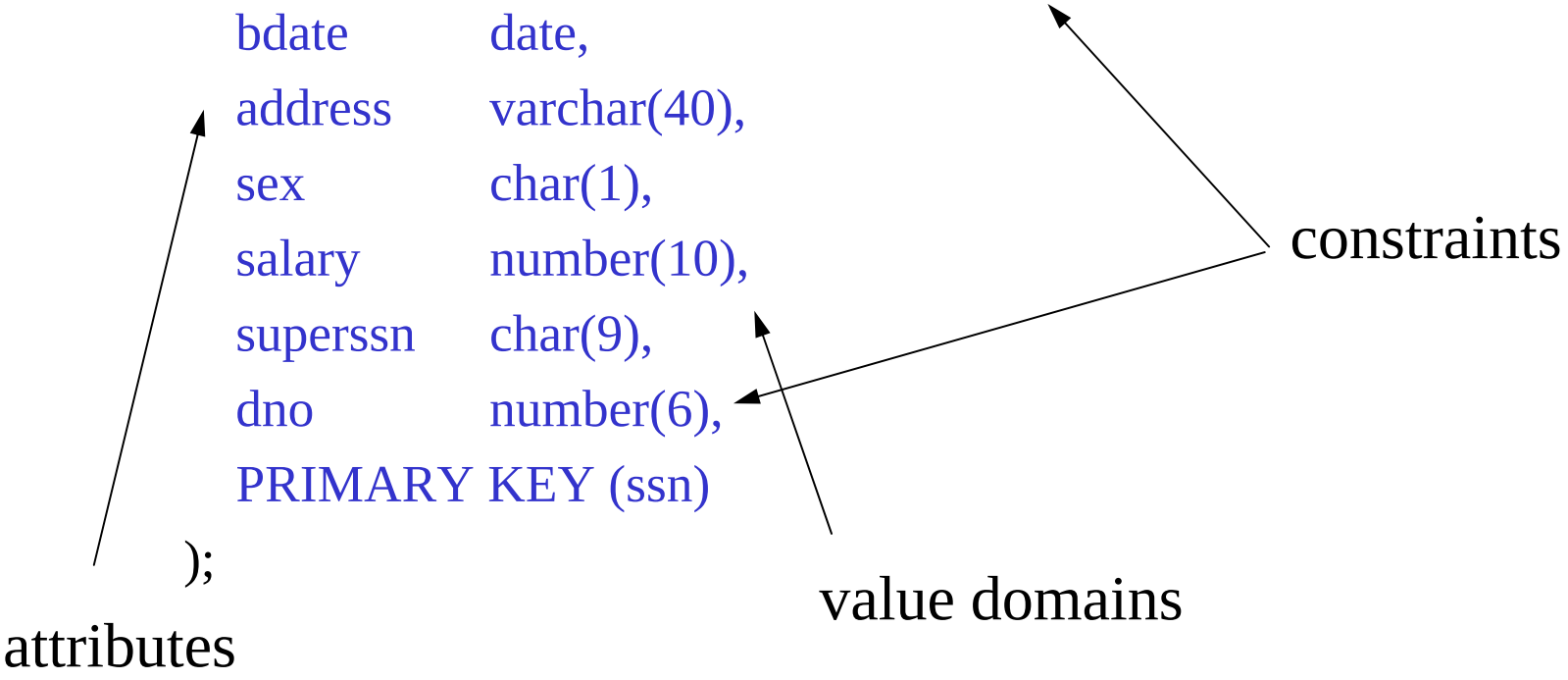
- Also, *named* value domains can be defined:

  CREATE DOMAIN ssn_type AS CHAR (9)

  (increases readability and improves maintainability)

Example: create employee

```
CREATE TABLE employee (
    name        varchar(20)     not null,
    ssn         char(9)         not null,
    bdate       date,
    address     varchar(40),
    sex         char(1),
    salary      number(10),
    superssn    char(9),
    dno         number(6),
    PRIMARY KEY (ssn)
);
```



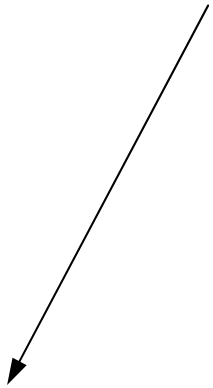
Constraints

```
CREATE TABLE employee (
   name        varchar(20)      not null,
   ssn         char(9)          not null,
   bdate       date,
   address     varchar(40),
   sex         char(1),
   salary      number(10),
   superssn    char(9),
   dno         number(6),

   PRIMARY KEY (ssn);

   CONSTRAINT employee_dno
   FOREIGN KEY (dno) REFERENCES  department(dnumber);
);
```

This constraints has no name

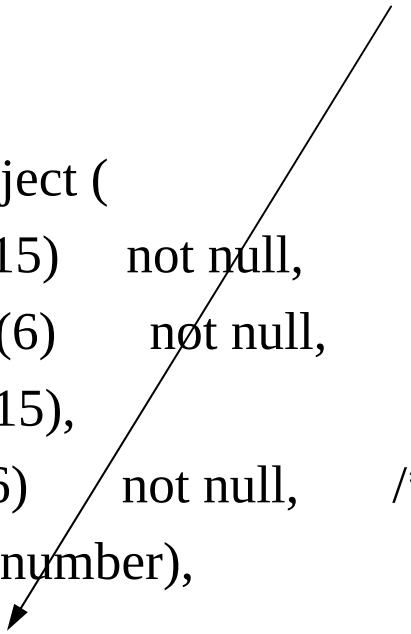
This constraint has a name

Composite keys



```
CREATE TABLE dept_locations (
   dnumber     number(6)       not null,
   dlocation   varchar(15)     not null,
   PRIMARY KEY (dnumber,dlocation)
   FOREIGN KEY (dnumber) REFERENCES  department(dnumber);
);
```

Other candidate keys



```
CREATE TABLE project (
   pname       varchar(15)     not null,
   pnumber     number(6)       not null,       /* primary key */
   plocation   varchar(15),
   dnum        number(6)       not null,       /* references department */
   PRIMARY KEY (pnumber),
   UNIQUE (pname)
);
```

Special Problem with foreign keys: what to do when database changes?

```
CREATE TABLE employee (
   ssn          char(9)          not null,
   ……...
   superssn     char(9),
   dno          number(6),

   PRIMARY KEY (ssn);
   CONSTRAINT employee_superssn
   FOREIGN KEY (superssn) REFERENCES  employee(ssn);

   CONSTRAINT employee_dno
   FOREIGN KEY (dno) REFERENCES  department(dnumber);
);
```

Special Problem with foreign keys: what to do when database changes?

```
CREATE TABLE employee (
    ssn             char(9)             not null,
    ……...
    superssn        char(9),
    dno             number(6),

    PRIMARY KEY (ssn);

    CONSTRAINT employee_superssn
    FOREIGN KEY (superssn) REFERENCES  employee(ssn)
          ON DELETE set null ON UPDATE cascade;
    CONSTRAINT employee_dno
    FOREIGN KEY (dno) REFERENCES  department(dnumber);
);
```

When the referenced tuple is deleted, the referring attribute will be set to null

When the referred attribute changes all references to it also change (not in Oracle 8, but in SQL92)

Special Problem with foreign keys: what to do when database changes?

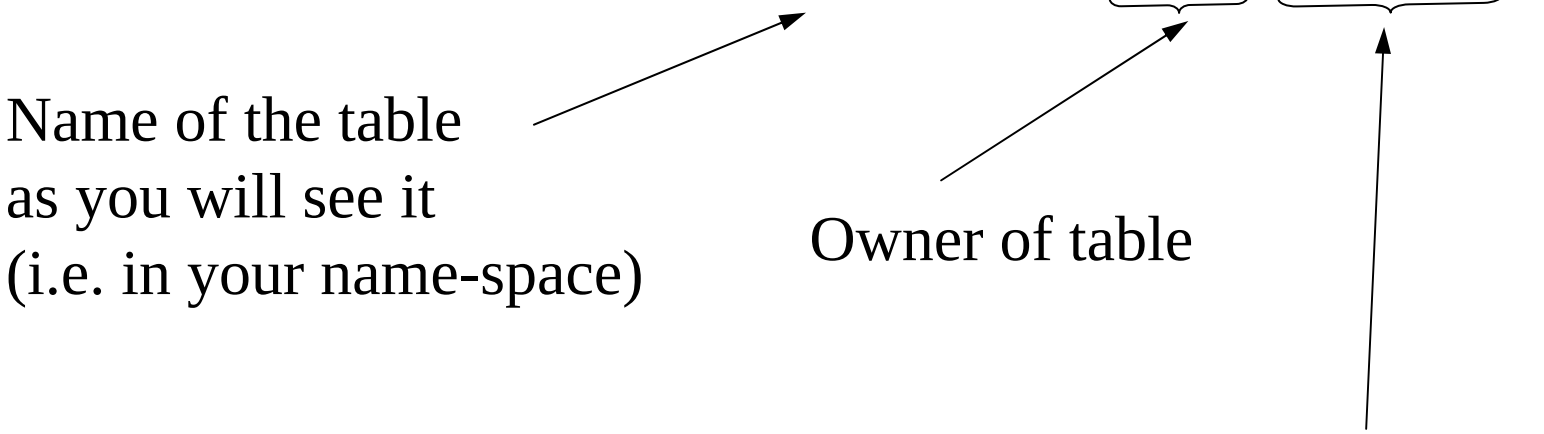
```
CREATE TABLE employee (
   ssn          char(9)          not null,
   ……...
   superssn     char(9),
   dno          number(6),

   PRIMARY KEY (ssn);

   CONSTRAINT employee_superssn
   FOREIGN KEY (superssn) REFERENCES  employee(ssn);

   CONSTRAINT employee_dno
   FOREIGN KEY (dno) REFERENCES  department(dnumber)
      ON DELETE cascade …… ;
);
```

When referred tuple is deleted all referring tuples are also deleted ('if a department is wound up, all employees are fired')

Seeing other peoples tables

- Create synonyms

```
CREATE SYNONYM employee FOR noran.employee;
```



Deleting schemas and tables

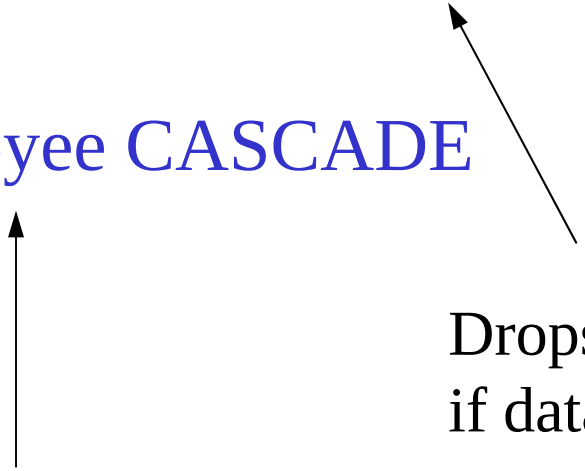
DROP SCHEMA company

DROP TABLE employee

Deleting schemas and tables

DROP SCHEMA company CASCADE

DROP TABLE employee CASCADE



Drops schema even
if database not empty

Drops table even if not empty

The end