• Our prior examples have assumed that tables existed
• We now describe how to define:
  – Tables
  – Derived Tables, i.e., Views
  – Table Constraints
Elementary SQL Data Types

CHAR(n) Fixed-length string of n characters

VARCHAR(n) Variable length string of up to n characters

INT, INTEGER, SHORTINT INT=INTEGER;
   SHORTINT may use less bits than INT

REAL, FLOAT, DOUBLE PRECISION
   REAL=FLOAT

DECIMAL(n,d) Has n digits, with d digits after the decimal point. E.g., 678.9 is of type DECIMAL(4,1)

DATE, TIME E.g., ’1997-05-19’, ’16:02:59’
CREATE TABLE

• In its basic form, CREATE TABLE is used to declare the following metadata:
  1. The name of the table
  2. The names and types of the attributes

• Example:
  CREATE TABLE STUDENT
  (SID INTEGER,
   NAME VARCHAR(255),
   MAJOR CHAR(3));
**Default Values**

- In CREATE TABLE, we can specify default values for the attributes.

- Example:
  ```sql
  CREATE TABLE STUDENT
  (SID INTEGER,
   NAME CHAR(255),
   MAJOR CHAR(3) DEFAULT '***');
  ```

- So, “INSERT INTO STUDENT(SID,NAME) VALUES(111,'Joe’)” will assign *** to MAJOR.

- If we don’t specify a default value, then the system uses NULL as the default.
The NOT NULL Constraint

• NOT NULL means that the value must be specified when the tuple is created

• Example:
  CREATE TABLE STUDENT
  (SID INTEGER NOT NULL,
   NAME CHAR(255),
   MAJOR CHAR(3));

• Useful for the attributes of a key, since a record won’t make sense if the key values are absent
Integrity Constraints on Individual Records

- The CREATE declaration can prohibit records that violate a boolean condition

- CREATE TABLE COURANT_STUDENT
  (SID INTEGER,
   NAME CHAR(255),
   MAJOR CHAR(3),
   CHECK (MAJOR IN ('CS', 'MATH'))
   CHECK (SID > 0 AND SID < 100000));
Declaring Supersets of Keys

- The constraint UNIQUE(A1, \ldots ,An) specifies that the attributes A1, \ldots ,An are a superset of a key.

- The system will ensure that no two rows will agree on all of the attributes A1, \ldots ,An.

- CREATE TABLE STUDENT
  (SID INTEGER,
   NAME CHAR(255),
   ZIP CHAR(5),
   ZIP_EXT CHAR(4),
   UNIQUE (SID),
   UNIQUE (ZIP,ZIP_EXT));
Primary Keys

- PRIMARY KEY (A1, ..., An) serves to:
  1. Declare A1, ..., An as UNIQUE
  2. Hint about how to organize the file
  3. Allow A1, ..., An to be part of a “foreign key” constraint

- CREATE TABLE STUDENT
  (SID INTEGER, ...
  PRIMARY KEY (SID),
  UNIQUE (ZIP,ZIP_EXT));

- There can only be one primary key
**Foreign Keys**

- When the schema of a relation $R$ contains attributes $A_1, \ldots, A_k$ that are the key of another relation $S$, then those attributes in $R$ are called a *foreign key*.

- $\text{STUDENT}(\text{SID}, \text{NAME}, \text{DEPT})$
  $\text{OFFICES}(\text{DEPT}, \text{ADDRESS})$

- Since $\{\text{DEPT}\}$ is included in schema($\text{STUDENT}$), and $\{\text{DEPT}\}$ is a key for $\text{OFFICES}$, then $\{\text{DEPT}\}$ is a foreign key from $\text{STUDENT}$ to $\text{OFFICES}$.

- A foreign key implies that for every tuple in $R$, there is a uniquely associated tuple in $S$. 
Foreign Key Constraints

- The constraint FOREIGN KEY (A1, \ldots ,An) will ensure that for each tuple with non-NULL values on A1, \ldots ,Ak, there must exist an associated tuple in the target relation.

- CREATE TABLE STUDENT
  ( \ldots DEPT CHAR(5), \ldots
  FOREIGN KEY (DEPT) REFERENCES OFFICES(DEPT));

- \{DEPT\} must be the primary key of OFFICES

- FOREIGN KEY enforces referential integrity
Maintaining Referential Integrity

• If an attempted operation would violate a referential integrity constraint, then the system must either:
  – Reject the operation, or
  – Accept the operation and make an adjustment to restore integrity

• If there is a foreign key constraint from $R$ to $S$, then integrity can be violated by deleting or updating tuples in $S$

• The ON DELETE and ON UPDATE options of FOREIGN KEY specify the adjustment policy
**Handling Deletions**

Suppose there is a foreign key from $R$ to $S$, and an attempt is made to delete a tuple $t$ from $S$. Then with ON DELETE we may specify:

**NO ACTION** The deletion of $t$ from $S$ is prohibited if $t$ is referenced by any tuple of $R$. This is the default.

**CASCADE** If it is possible to delete all tuples in $R$ that refer to $t$, then they are deleted and $t$ is deleted.

**SET NULL** Then $t$ is deleted, and all tuples that refer to $t$ have their foreign-key attributes set to NULL.

**SET DEFAULT** Similar to SET NULL.
**A FOREIGN KEY WITH CASCADED DELETES**

CREATE TABLE STUDENT(
    SID INTEGER,
    NAME CHAR(255),
    DEPT CHAR(5),
    MAJOR CHAR(4),

    FOREIGN KEY (DEPT)
    REFERENCES OFFICES(DEPT)
    ON DELETE CASCADE

);
HANDLING UPDATES

Suppose there is a foreign key from $R$ to $S$, and an attempt is made to change the primary key values for a tuple $t$ in $S$. Then with ON UPDATE we may specify:

**NO ACTION** The update of $t$ in $S$ is prohibited if $t$ is referenced by any tuple of $R$. This is the default.

**CASCADE** All tuples in $R$ that refer to $t$ have their foreign key values updated to the new values.

**SET NULL** All tuples in $R$ that refer to $t$ have their foreign-key attributes set to NULL.

**SET DEFAULT** Similar to SET NULL.
Views are essentially macros. The tables specified by the views don’t exist, but the database management system will create the table whenever the user queries the view.

CREATE VIEW blockbuster AS SELECT * FROM movie_titles WHERE revenue > 100000000
GRANT

Grant is a form of discretionary access control.

GRANT SELECT ON movie_titles TO movie_clerk
GRANT ALL PRIVILEGES ON movie_titles TO costner
One can in principle add where clauses to GRANT statements, but the performance is ruinous.
Data Definition Exercises

1. Define a relation ACTOR with fields NAME and INCOME.

2. Which foreign key dependencies would be useful between ACTOR and STARRING.

3. Define a view ACTORREVENUE with fields MOVIE_NAME, ACTOR_NAME, and REVENUE that relates actors to films in which they are starring.
1. Write a query that discovers which actor draws an average revenue of 30 million dollars or more in movies in which he or she stars.

2. Write a query that finds promising starring pairs of actors. (First decide what this vague specification may mean.)
NULLs

Assume that we do not have the value of p# for some customers. Then, the special value of NULL is assigned there by the system.

Find all cnames for customers for whom we do not have a plant assigned yet

SELECT CNAME FROM CUSTOMERS WHERE (P# IS NULL);

Generally, NULL stands for value unknown, so all NULLs are considered different by the system.