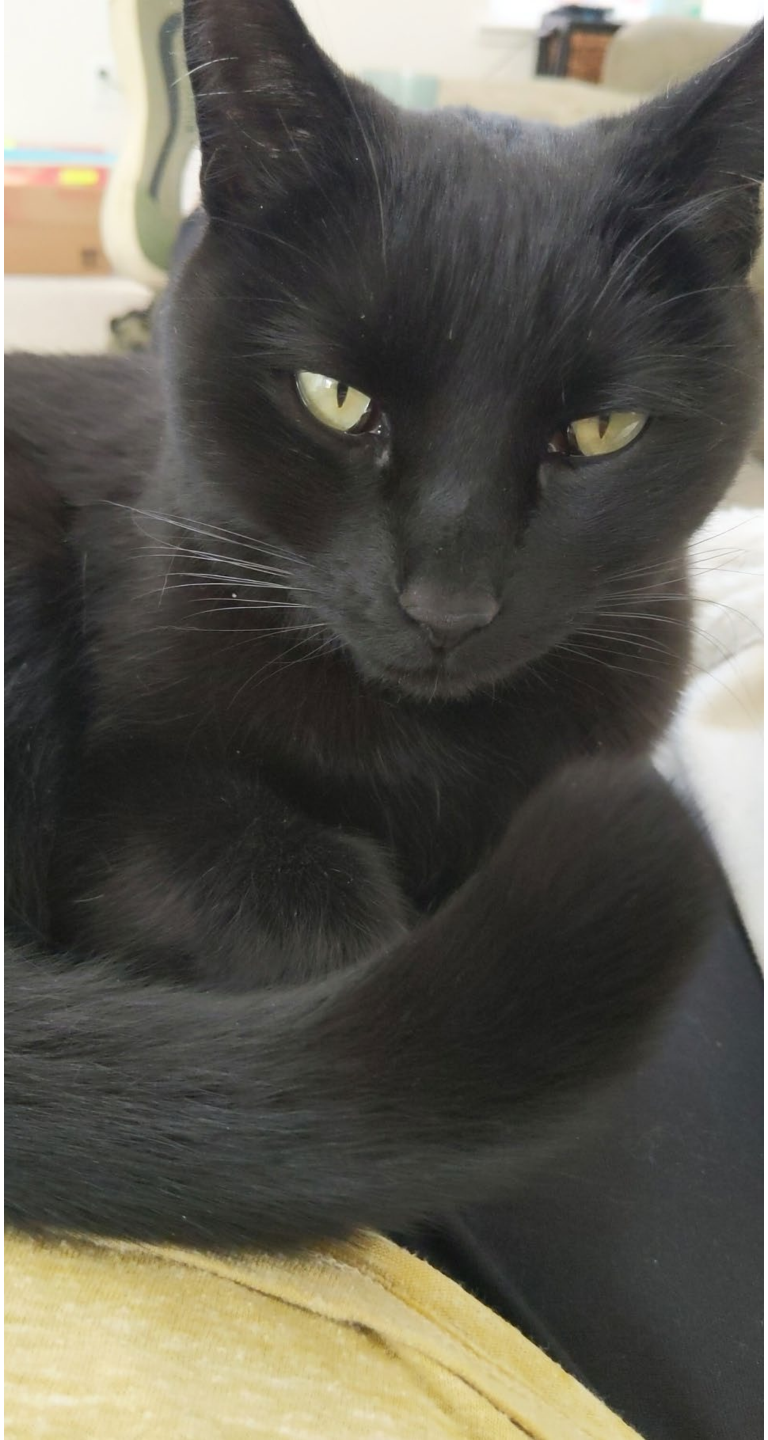


Design

ERIN KEITH

Goals

1. Background
2. Introduce the Relational Model
3. Practice!

There is a student in our class who is requesting a note-taker.

- The pay for providing notes is $100 stipend toward next semester's tuition and fees. Those interested in the notetaking position should complete "note-taker application" @ https://www.unr.edu/drc/student-application.
- If you have additional questions or need assistance, contact Geoff Kettling (kettling@unr.edu).

iMDb

Example

in-class Movie Database

- What does it mean to be a movie?

General Data Model

OOP

State
- attributes

Behavior
- functions

Constraints
- types

RDBMS

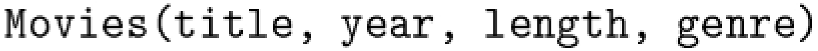
State
- fields

Behavior
- query, insert, update

Constraints
- types, keys

Schemas

The name of a relation and the set of attributes for a relation is called the **schema** for that relation.

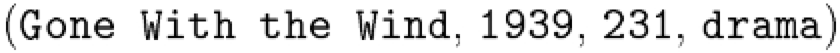


```
Movies(title, year, length, genre)
```

- A database consists of one or more relations. The set of schemas for the relations of a database is called a **database schema**.

Tuples

The rows of a relation, other than the header row containing the attribute names, are called **tuples**.

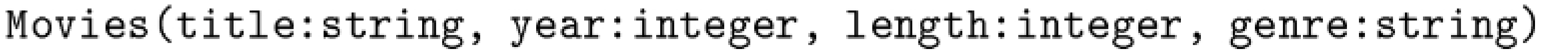


(Gone With the Wind, 1939, 231, drama)

- a tuple has one component for each attribute of the relation
- when a tuple appears in isolation, the attributes do not appear

Domains

each component of each tuple be some elementary type such as integer or string.



```
Movies(title:string, year:integer, length:integer, genre:string)
```

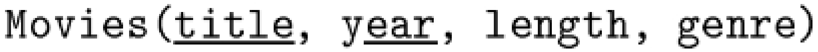
- it is possible to include the **domain**, or data type, for each attribute in a relation schema

Relations

- order of tuples doesn’t matter
- relations change
  - insert new tuple
  - delete existing tuple
  - update existing tuple
- schemas are ***not*** expected to change!

Keys

a set of attributes forms a **primary key** for a relation

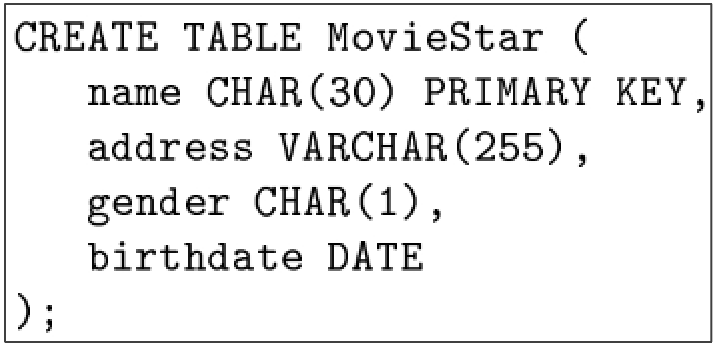


$\texttt{Movies(}\underline{\texttt{title}}\texttt{, }\underline{\texttt{year}}\texttt{, length, genre)}$

- no two tuples in a relation instance to have the same values in all the attributes of the key

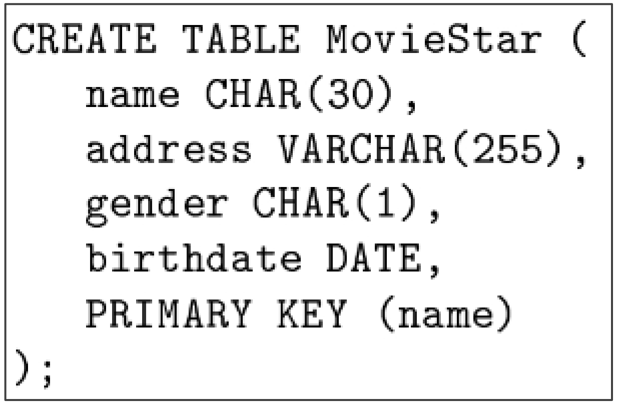
Keys

- cannot be **NULL**
- the set of keys must be *unique*
  - otherwise the insert operation will result in an error



```
CREATE TABLE MovieStar (
    name CHAR(30) PRIMARY KEY,
    address VARCHAR(255),
    gender CHAR(1),
    birthdate DATE
);
```

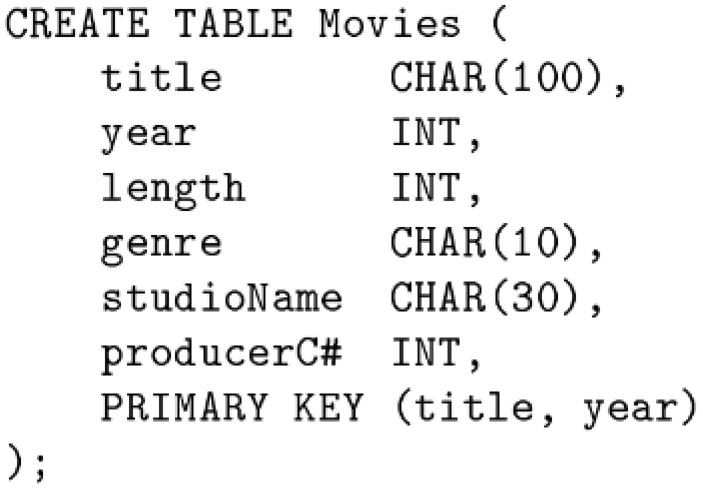
OR



```
CREATE TABLE MovieStar (
    name CHAR(30),
    address VARCHAR(255),
    gender CHAR(1),
    birthdate DATE,
    PRIMARY KEY (name)
);
```

Keys

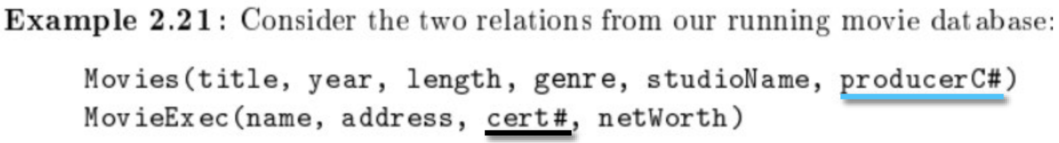
- cannot be **`NULL`**
- the set of keys must be *unique*
  - otherwise, the insert operation will result in an error



```
CREATE TABLE Movies (
    title        CHAR(100),
    year         INT,
    length       INT,
    genre        CHAR(10),
    studioName   CHAR(30),
    producerC#   INT,
    PRIMARY KEY (title, year)
);
```

Foreign Key

- cannot be **`NULL`**
- indicates a dependency on another relation’s primary key

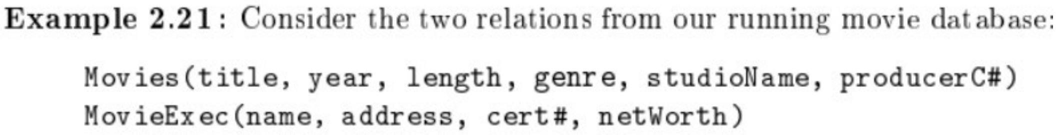


Example 2.21: Consider the two relations from our running movie database:

```
Movies(title, year, length, genre, studioName, producerC#)
MovieExec(name, address, cert#, netWorth)
```

Foreign Key

- referential integrity constraint
  - a value appearing in one context also appears in another, related context



Example 2.21: Consider the two relations from our running movie database:

```
Movies(title, year, length, genre, studioName, producerC#)
MovieExec(name, address, cert#, netWorth)
```

Practice

- What does it mean to be a ___?

Entity Relationship Diagram

Model database schemas

- Entity

Entity		
Key	Field	Type
Key	Field	Type
Key	Field	Type

Entity Relationship Diagram

Model database schemas

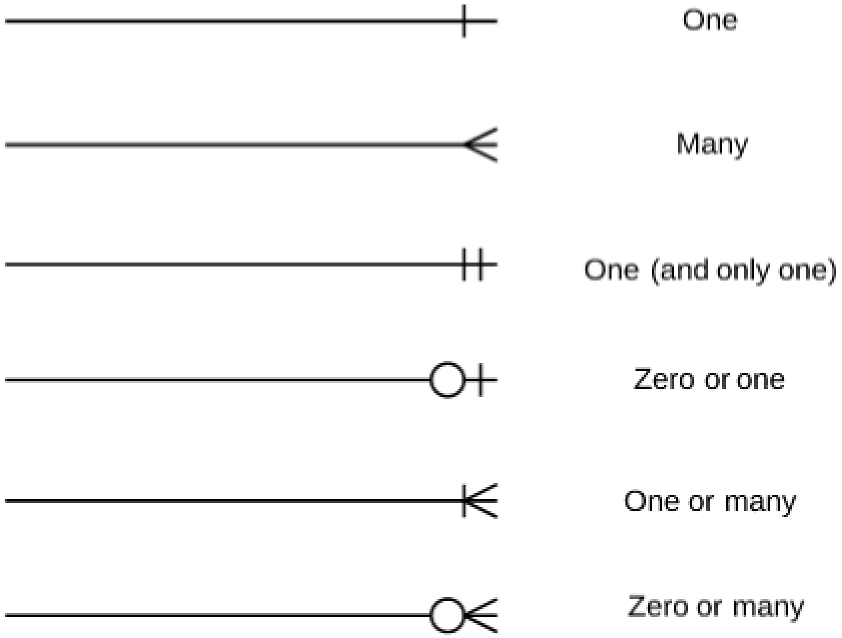
- Entity (table schema)
  - Key
    - Primary Key: PK
    - Foreign Key: FK
  - Field (column)
  - Type (datatype)

Movie		
PK	title	varchar(255)
PK	year	int
	length	int
	genre	varchar(255)
	studioName	varchar(255)
FK	producerC#	int

Entity Relationship Diagram

Model database schemas

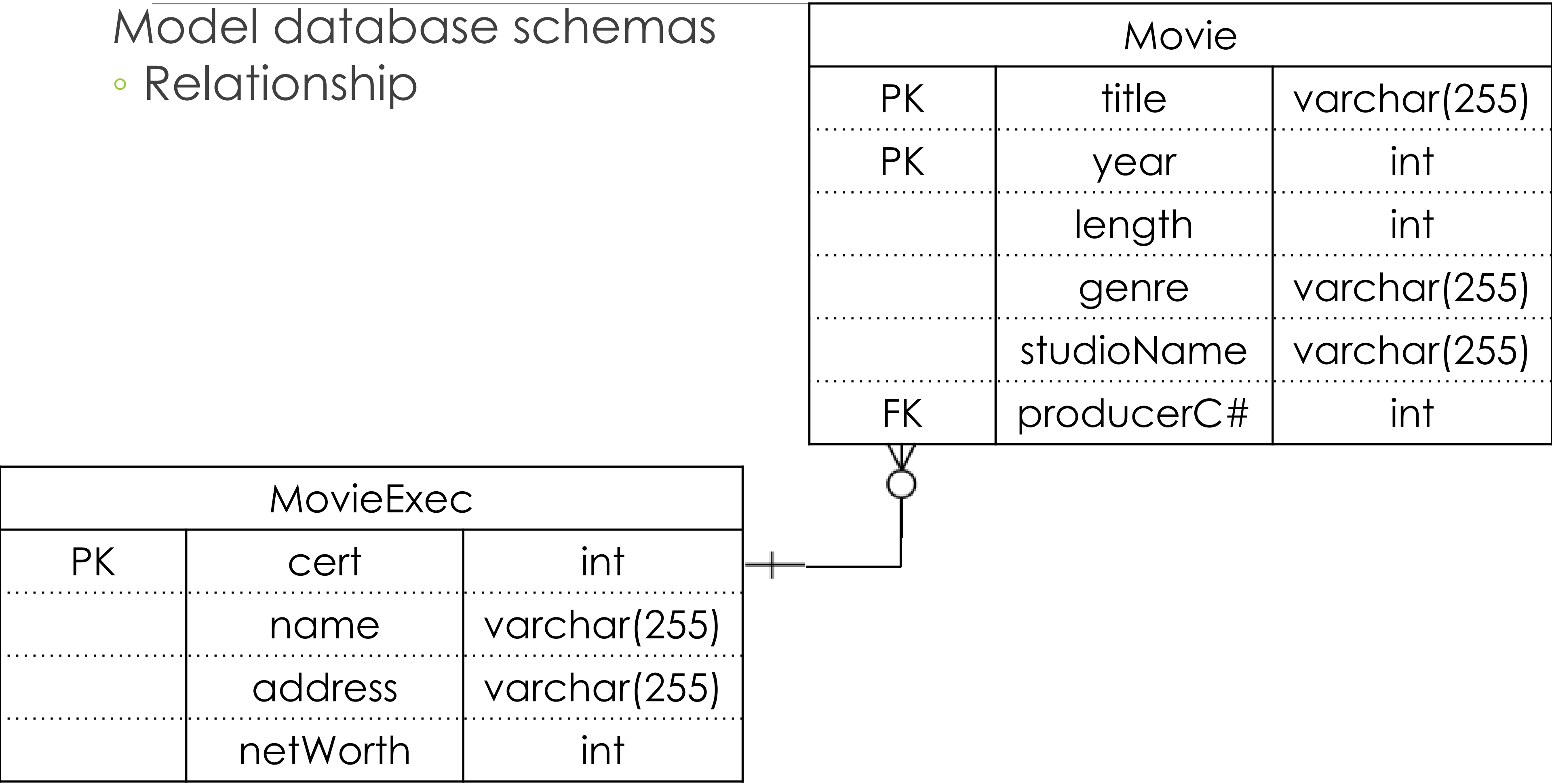
- Relationship
  - **Cardinality**: the maximum number of times an instance in one entity can relate to instances of another entity
  - **Ordinality**: the minimum number of times an instance in one entity can be associated with an instance in the related entity.



Entity Relationship Diagram

Model database schemas

- Relationship



Movie		
PK	title	varchar(255)
PK	year	int
	length	int
	genre	varchar(255)
	studioName	varchar(255)
FK	producerC#	int

MovieExec		
PK	cert	int
	name	varchar(255)
	address	varchar(255)
	netWorth	int

Next Class

Module:

Week 3: Design, Ch 3

Topic:

Dependencies