The Relational Model

Sections to read:

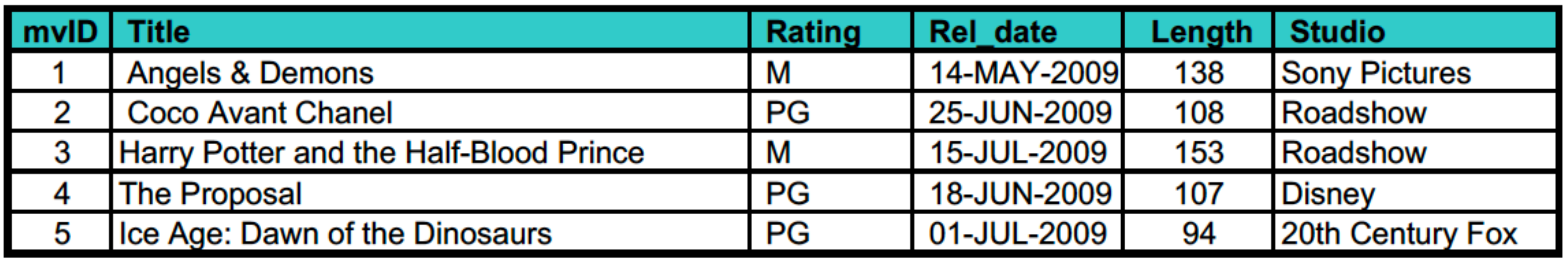
- “Basics of the Relational Model”
- “Keys and Foreign Keys”
- ”Defining a relation schema in SQL”

Data models

- A data model is a language for describing data or information. The description generally consists of three parts:
  - Structure of the data, or data structure.
  - Operations on the data.
  - Constraints on the data
- The relational data model is the most popular data model. It is for managing structured data.

The relational model

- The relational model is based on a single *structure* to represent data, a two-dimensional table called a *relation*.
- Example -- the “movie” relation.



mvID	Title	Rating	Rel_date	Length	Studio
1	Angels & Demons	M	14-MAY-2009	138	Sony Pictures
2	Coco Avant Chanel	PG	25-JUN-2009	108	Roadshow
3	Harry Potter and the Half-Blood Prince	M	15-JUL-2009	153	Roadshow
4	The Proposal	PG	18-JUN-2009	107	Disney
5	Ice Age: Dawn of the Dinosaurs	PG	01-JUL-2009	94	20th Century Fox

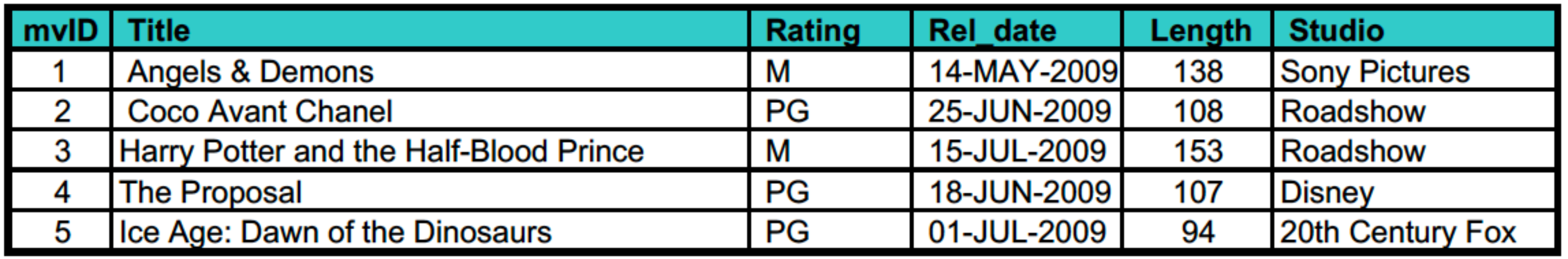
Basics of the relational model

- **Attributes** are the names for the columns of a relation. Each attribute has a domain --- the set of all possible values allowed for the attribute.
- **Tuples** are the rows of data in a relation comprising components that are atomic --- a component cannot be broken into smaller components. Each component of a tuple is a value from the corresponding attribute domain.
- **Schema** of a relation is the name of a relation and its attributes. The schema for a relation is written as relation name followed by a list of is attributes in a pair of brackets.

Basics of the relational model …

- A **relation instance** is a set of tuples, or the data, for a given relation.
  - It is common to update the content of a relation and have a new relation instance.
  - It is less common to update the schema.
- Typically, a database system only keeps the ”current” relation instance – the data “now” kept in relations.
- The term “relation” can refer to a relation schema or a relation instance, and usually can be determined from the context.

The Movie relation: an example



mvID	Title	Rating	Rel_date	Length	Studio
1	Angels & Demons	M	14-MAY-2009	138	Sony Pictures
2	Coco Avant Chanel	PG	25-JUN-2009	108	Roadshow
3	Harry Potter and the Half-Blood Prince	M	15-JUL-2009	153	Roadshow
4	The Proposal	PG	18-JUN-2009	107	Disney
5	Ice Age: Dawn of the Dinosaurs	PG	01-JUL-2009	94	20th Century Fox

- The schema of the Movie relation is Movie(mvID, Title, Rating, Rel_date, Length, Studio).
- The domain of mvID is Integer (data type).
- The current instance of Movie relation has 5 tuples. The first tuple of Movie relation has 6 components:

  - 1, ‘Angels & Demons’, M, 14-05-2009, 138, ‘Sony Pictures’.

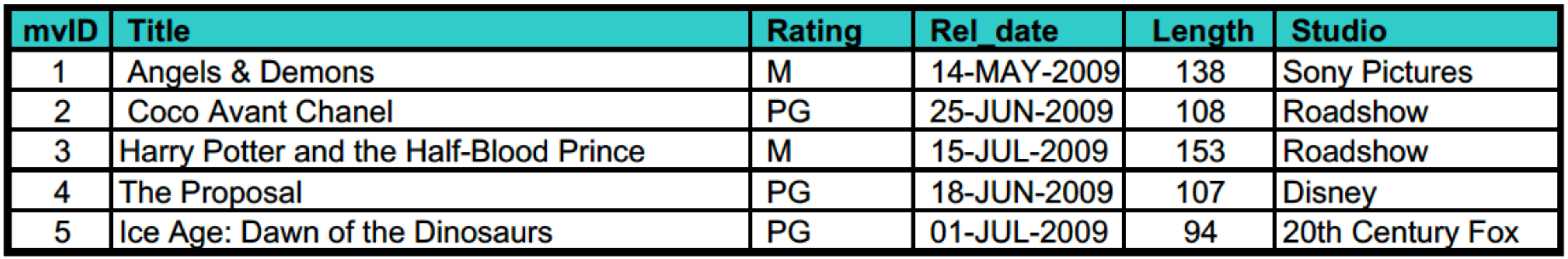
  Note that the date ’14-05-2009’ is an atomic value.

Data integrity constraints – Key and Primary Key

- An attribute or list of attributes are a *Key* of a relation if no two tuples of the relation may agree in all the attribute(s) on the list. If there are several candidate keys, one is specified as the *Primary Key*.
  - No two movies can have the same value for *mvID*. *mvID* is the artificial key of Movie.
  - A movie (represented by *mvID*) can be linked to multiple genres, and a genre can link to multiple movies. So the *mvID* and genre together – {*mvID, Genre*} -- are the only key for Classification.
- Primary key attributes are underlined in relation schemas.

Key and Primary Key …

- Note that the key constraint is a natural property of real-world data and part of a relation schema. Key definition can only be derived from the real-world. Although sample data in a relation can help disapprove a possible key definition, they cannot be used to prove a key definition.
- Example: For the Movie relation, even though the current content of the Movie relation may suggest that each movie has a distinct title, title is not a key for the relation. “No two movies can have the same title” is not a property that is always true for all movies to keep in a database.



mvID	Title	Rating	Rel_date	Length	Studio
1	Angels & Demons	M	14-MAY-2009	138	Sony Pictures
2	Coco Avant Chanel	PG	25-JUN-2009	108	Roadshow
3	Harry Potter and the Half-Blood Prince	M	15-JUL-2009	153	Roadshow
4	The Proposal	PG	18-JUN-2009	107	Disney
5	Ice Age: Dawn of the Dinosaurs	PG	01-JUL-2009	94	20th Century Fox

Data Integrity Constraints -- Foreign Keys

- The **foreign key** constraint declares that an attribute or attributes of a relation references the primary key or key of a second relation.
  - Values for the foreign key in the first relation must appear in the referenced attributes of the second relation.
  - A child-parent relationship -- the first and second relations can be seen as “child” and “parent” relations.
- Foreign keys in a relation schema are usually denoted by an asterisk (*).

The Foreign-Key Constraint …

- A foreign-key constraint on a relation *prevents* introducing nonsensical data into the relation.
  - No insertion or update to the child relation introduces values not found in the parent relation.
  - No deletion or update to the parent relation causes some tuples of the child relation to “dangle.”

Relation schemas with data integrity constraints

- Primary key attributes are underlined.
  - The primary key of a relation may comprise more than one attribute.
- Foreign key attributes are annotated with the asterisk.
- A primary key attribute can also be a foreign key.
- Examples:

Movie(mvID, Title, Rating, Rel_date, Length, Studio)
Classification(mvID*, genre)

Relational databases

- A relational database schema is the set of all relation schemas in the database.
- A relational database instance comprises a set of relation instances. Or informally, a relational database comprises a set of relations, or tables.

A (small) Movies database

- Data to keep in the database:
  - Movie: mvID, title, rating, release date, length (in minutes), producer studio. mvID is an artificial attribute (not natural information).
  - Classification: a movie may be associated with several genres (comedy, action, violence, adventure, drama).
  - Cast: A movie has many actors, and an actor can perform in many movies.
  - Direct: A movie can have several directors (usually less than three) and a director can direct many movies.

The Movies database schema

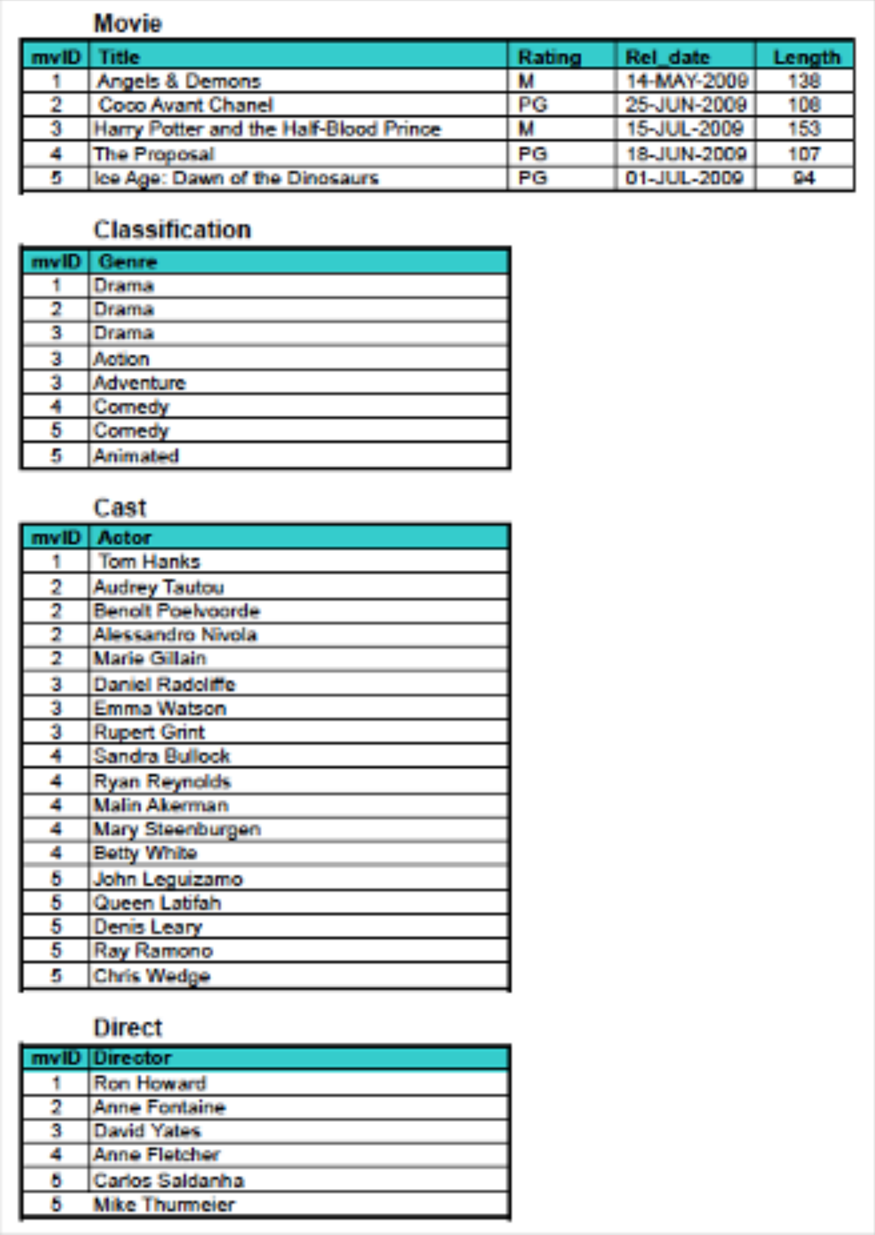
Movie(mvID, Title, Rating, Rel_date, Length, Studio)

Classification(mvID*, Genre)

Cast(mvID*, Actor)

Direct(mvID*, Director)

The (Small) Movies database



Movie

mvID	Title	Rating	Rel_date	Length
1	Angels & Demons	M	14-MAY-2009	138
2	Coco Avant Chanel	PG	25-JUN-2009	108
3	Harry Potter and the Half-Blood Prince	M	15-JUL-2009	153
4	The Proposal	PG	18-JUN-2009	107
5	Ice Age: Dawn of the Dinosaurs	PG	01-JUL-2009	94

Classification

mvID	Genre
1	Drama
2	Drama
3	Drama
3	Action
3	Adventure
4	Comedy
5	Comedy
5	Animated

Cast

mvID	Actor
1	Tom Hanks
2	Audrey Tautou
2	Benoît Poelvoorde
2	Alessandro Nivola
2	Marie Gillain
3	Daniel Radcliffe
3	Emma Watson
3	Rupert Grint
4	Sandra Bullock
4	Ryan Reynolds
4	Malin Akerman
4	Mary Steenburgen
4	Betty White
5	John Leguizamo
5	Queen Latifah
5	Denis Leary
5	Ray Ramono
5	Chris Wedge

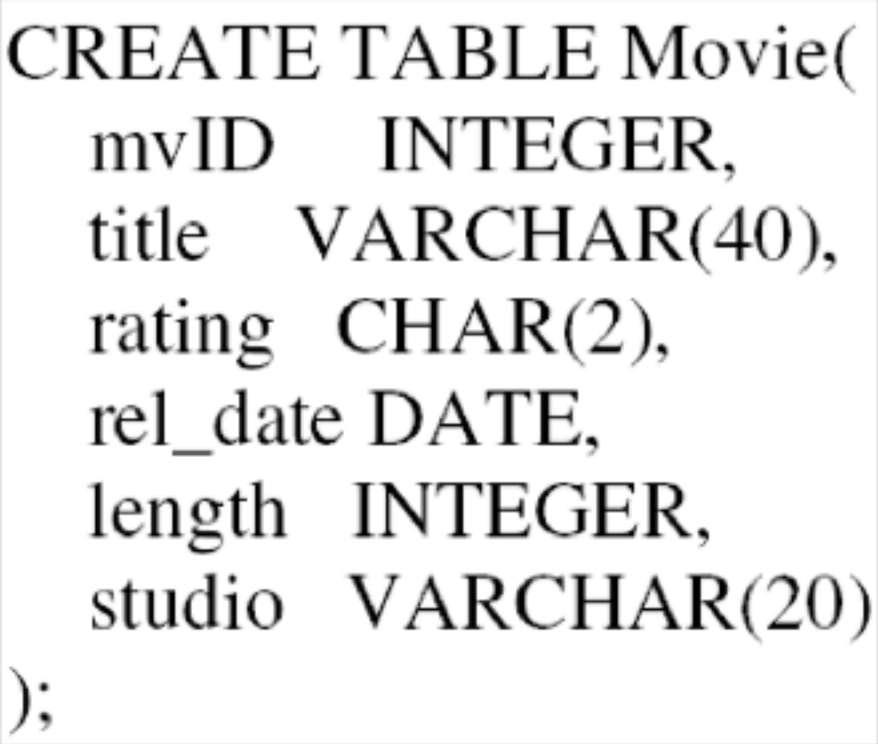
Direct

mvID	Director
1	Ron Howard
2	Anne Fontaine
3	David Yates
4	Anne Fletcher
5	Carlos Saldanha
5	Mike Thurmeier

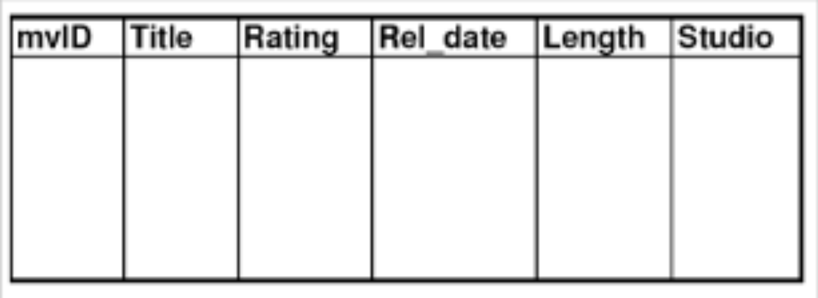
Define relation schemas in SQL

- SQL is a data definition language for defining database schemas and constraints.
  - The CREATE TABLE statement defines and creates an empty table.

The CREATE TABLE statement



```
CREATE TABLE Movie(
   mvID     INTEGER,
   title    VARCHAR(40),
   rating   CHAR(2),
   rel_date DATE,
   length   INTEGER,
   studio   VARCHAR(20)
);
```



mvID	Title	Rating	Rel_date	Length	Studio

Elements of Table Definition

- Most basic element: an attribute and its type.
- The most common types are:
  - INT or INTEGER (synonyms).
  - REAL or FLOAT (synonyms).
  - CHAR(*n*) = fixed-length string of *n* characters.
  - VARCHAR(*n*) = variable-length string of up to *n* characters.

SQL Values

- Strings require single quotes.
  - E.g., ‘Tom Hanks’, and ‘The Proposal’.
  - Two single quotes = real quote, e.g., ’`Mary’’s little lamb`’.
- Any value can be NULL.
  - NULL can have several semantics:
    - Value unknown, inapplicable, or withheld.

SQL Values …

- DATE and TIME are types in SQL and so there are DATE and TIME values.
- The default form of a date value is ’DD-MON-YYYY’ (e.g. ‘25-JUL-2010’).
- The form of a time value is: ’hh:mm:ss’, with an optional decimal point and fractions of a second following.
  - `TIME ’15:30:02.5’` = two and a half seconds after 3:30PM.

SQL Values …

- The SQL code below creates an empty table.

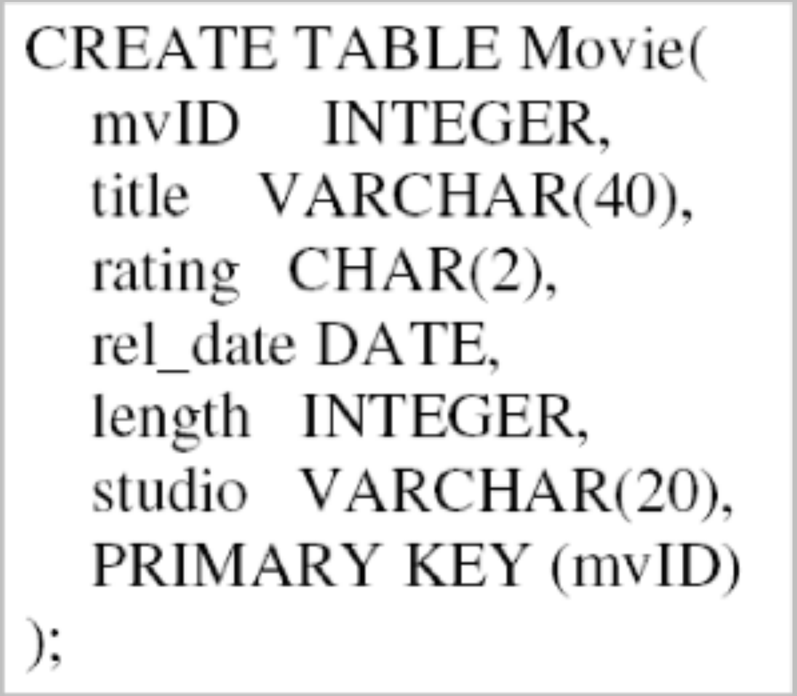
```
create table movie (
 mvid integer,
 title varchar(40),
 rating char(2),
 rel_date date,
 length integer,
 studio varchar(20));
```

- The code below insert data values into the table.

```
insert into movie values(1, ‘Mary’’s little lamb’, ‘G’, ’14-MAY-
2001’, 80, ‘Farmer’);
```

Defining Primary Keys in SQL

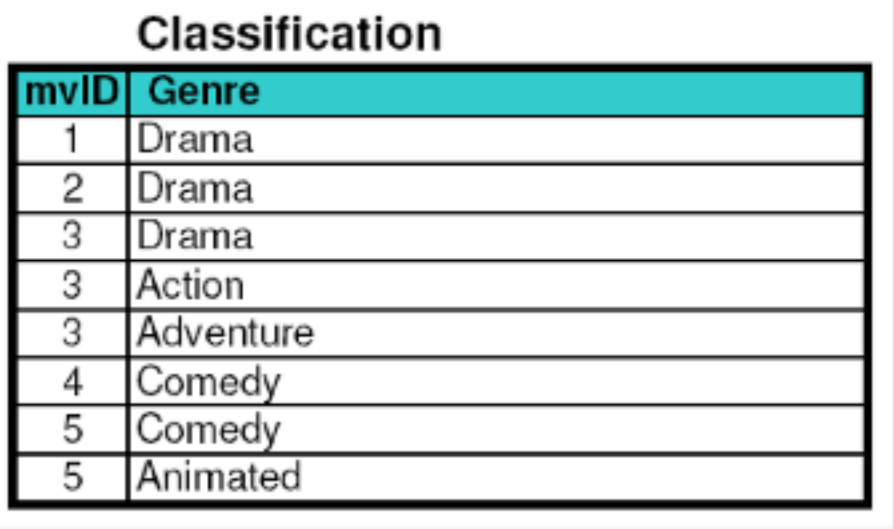
- A primary key definition PRIMARY KEY (…) is another element in the list of elements of a CREATE TABLE statement.
  - Movie(<u>mvID</u>, title, rating, length, studio)



```
CREATE TABLE Movie(
   mvID     INTEGER,
   title    VARCHAR(40),
   rating   CHAR(2),
   rel_date DATE,
   length   INTEGER,
   studio   VARCHAR(20),
   PRIMARY KEY (mvID)
);
```

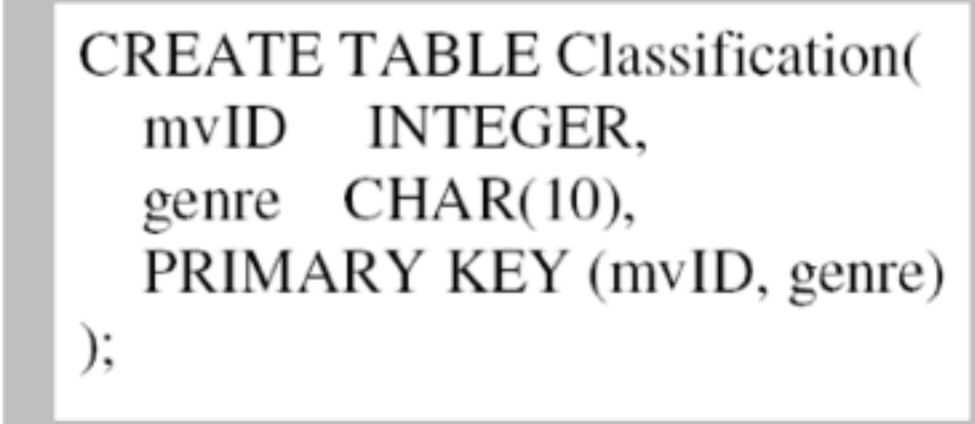
Defining Primary keys …

- Several attributes together make up a key.
  - Classification (mvID, Genre)



Classification

mvID	Genre
1	Drama
2	Drama
3	Drama
3	Action
3	Adventure
4	Comedy
5	Comedy
5	Animated



```
CREATE TABLE Classification(
   mvID    INTEGER,
   genre   CHAR(10),
   PRIMARY KEY (mvID, genre)
);
```

Ensuring Data Integrity: primary key unique constraints

- The unique constraint: tuples can not have the same value for all PK attributes.
- With the given Movie relation schema:

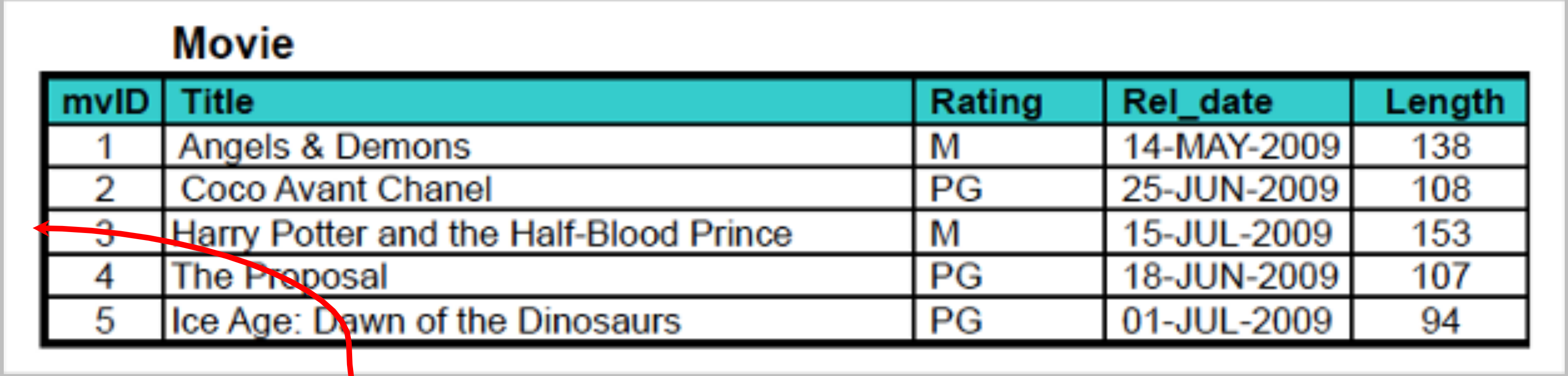
  Create table Movie (.. Primary key (mvID));

Underline indicates primary key!

Movie(mvID, Title, Rating, Rel_date, Length, Studio)

- mvID is the primary key for the Movie relation.

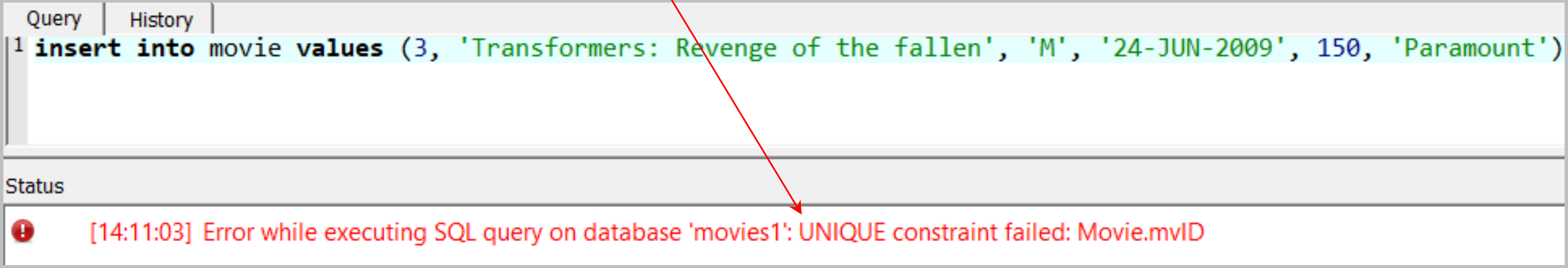
The unique constraint …



Movie

mvID	Title	Rating	Rel_date	Length
1	Angels & Demons	M	14-MAY-2009	138
2	Coco Avant Chanel	PG	25-JUN-2009	108
3	Harry Potter and the Half-Blood Prince	M	15-JUL-2009	153
4	The Proposal	PG	18-JUN-2009	107
5	Ice Age: Dawn of the Dinosaurs	PG	01-JUL-2009	94

Unique constraint violated!



```
insert into movie values (3, 'Transformers: Revenge of the fallen', 'M', '24-JUN-2009', 150, 'Paramount')
```

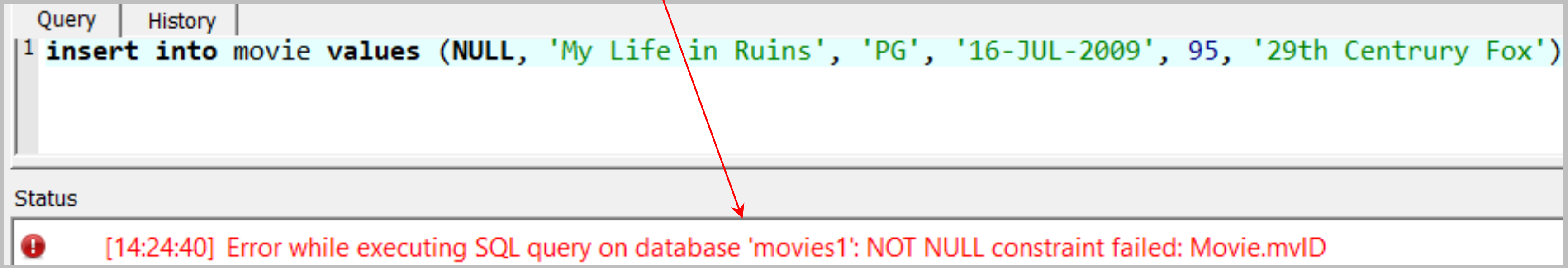
```
[14:11:03] Error while executing SQL query on database 'movies1': UNIQUE constraint failed: Movie.mvID
```

Ensuring Data Integrity: primary key entity integrity constraints

- The entity integrity constraint: Null is not allowed for PK attributes.

insert into movie
    values (NULL, 'My Life in Ruins', 'PG', '16-JUL-2009', 95, '29th
    Centrury Fox');

Entity integrity constraint violated!



```
insert into movie values (NULL, 'My Life in Ruins', 'PG', '16-JUL-2009', 95, '29th Centrury Fox')
```

```
[14:24:40] Error while executing SQL query on database 'movies1': NOT NULL constraint failed: Movie.mvID
```

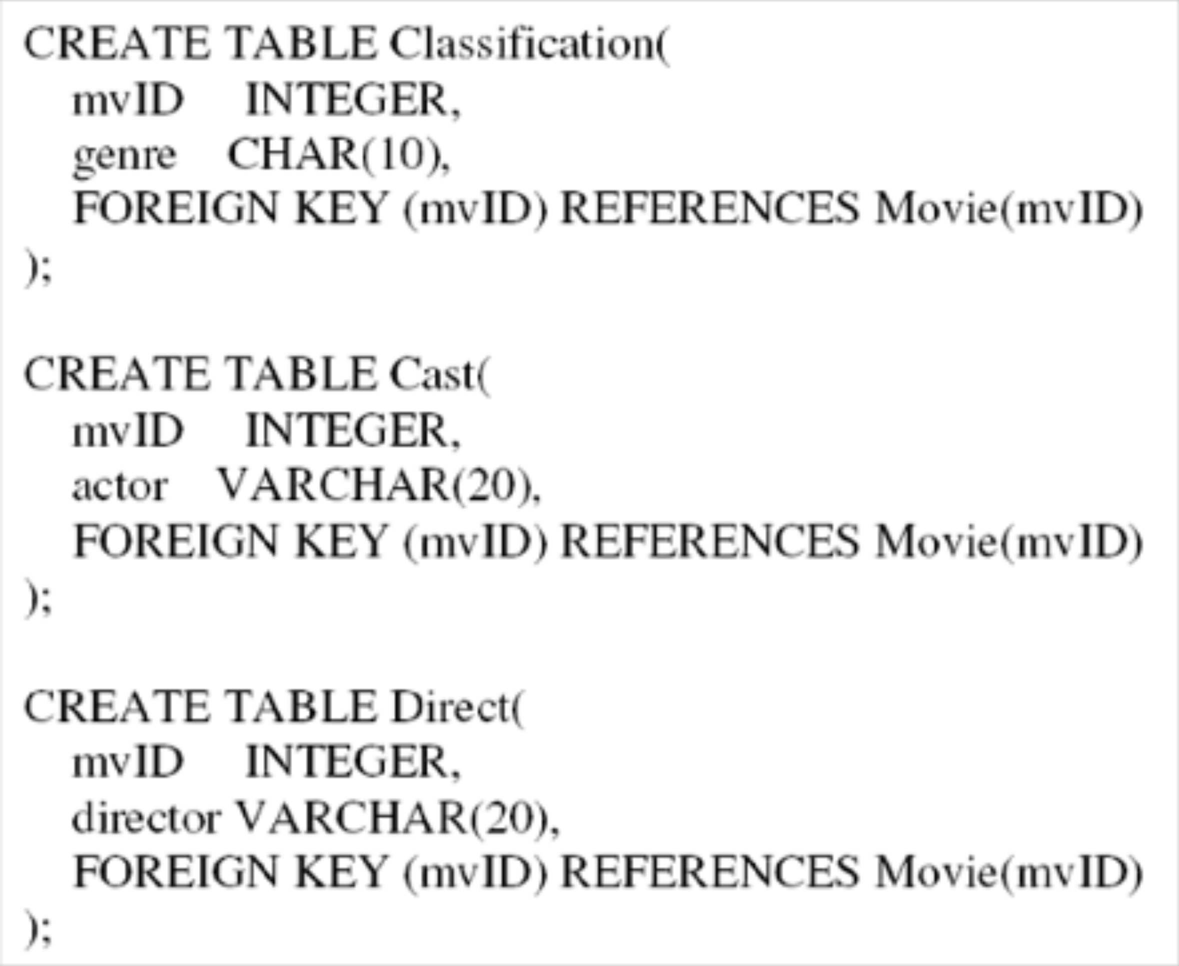
Defining Foreign Keys

- A foreign key definition

  FOREIGN KEY (….) REFERENCES
  relation(attributes)

  can be added as an element in the CREATE TABLE statement.

Defining Foreign Keys …

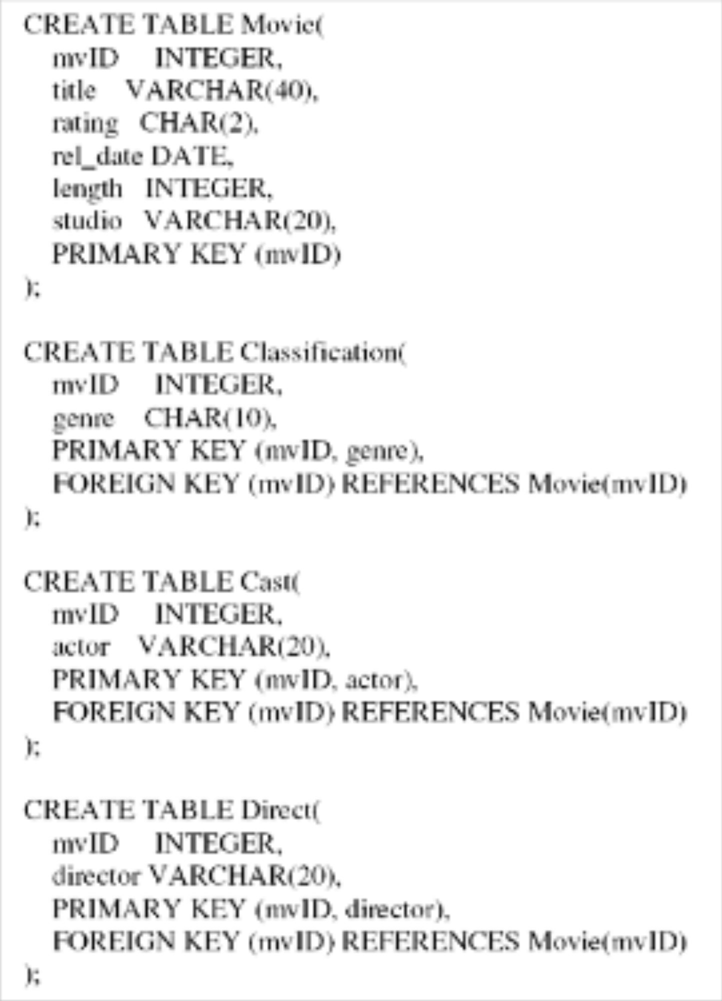


```
CREATE TABLE Classification(
   mvID     INTEGER,
   genre    CHAR(10),
   FOREIGN KEY (mvID) REFERENCES Movie(mvID)
);

CREATE TABLE Cast(
   mvID     INTEGER,
   actor    VARCHAR(20),
   FOREIGN KEY (mvID) REFERENCES Movie(mvID)
);

CREATE TABLE Direct(
   mvID     INTEGER,
   director VARCHAR(20),
   FOREIGN KEY (mvID) REFERENCES Movie(mvID)
);
```

The Movies DB – Primary and Foreign Keys in SQL



```
CREATE TABLE Movie(
   mvID    INTEGER,
   title   VARCHAR(40),
   rating  CHAR(2),
   rel_date DATE,
   length  INTEGER,
   studio  VARCHAR(20),
   PRIMARY KEY (mvID)
);

CREATE TABLE Classification(
   mvID    INTEGER,
   genre   CHAR(10),
   PRIMARY KEY (mvID, genre),
   FOREIGN KEY (mvID) REFERENCES Movie(mvID)
);

CREATE TABLE Cast(
   mvID    INTEGER,
   actor   VARCHAR(20),
   PRIMARY KEY (mvID, actor),
   FOREIGN KEY (mvID) REFERENCES Movie(mvID)
);

CREATE TABLE Direct(
   mvID    INTEGER,
   director VARCHAR(20),
   PRIMARY KEY (mvID, director),
   FOREIGN KEY (mvID) REFERENCES Movie(mvID)
);
```

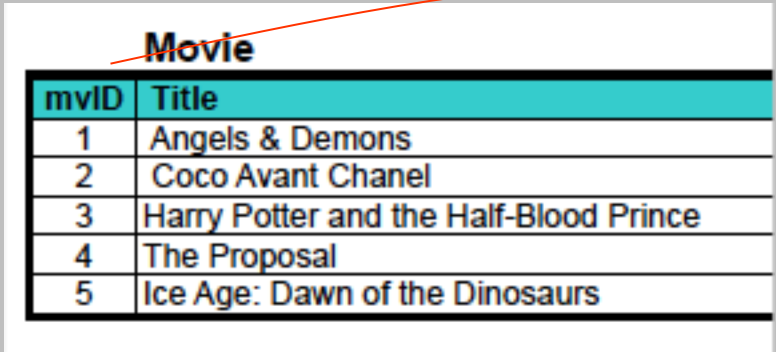
The Movies DB – Primary and Foreign Keys …

- A movieID that does not appear in the Movie relation cannot be added into the Classification, Cast or Direct relations.
- A tuple (describing a movie) can only be deleted from the Movie relation if its mvID does not appear in Classification, Cast or Direct. In other words, dangling mvID values in Classifiction, Cast or Direct are not allowed.

Ensuring Data Integrity – foreign keys

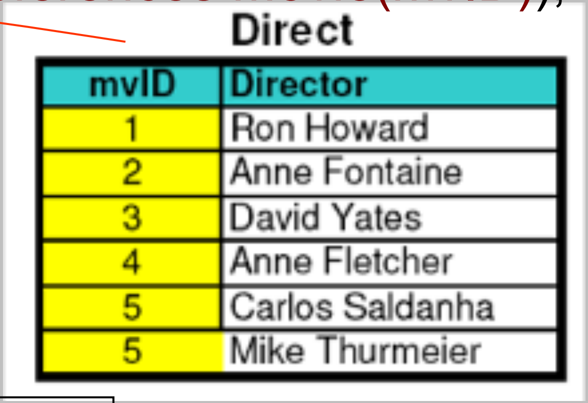
- The foreign key referential integrity constraint ensures sensible insertion.

Create table Direct(... foreign key (mvID) references Movie(mvID));



Movie

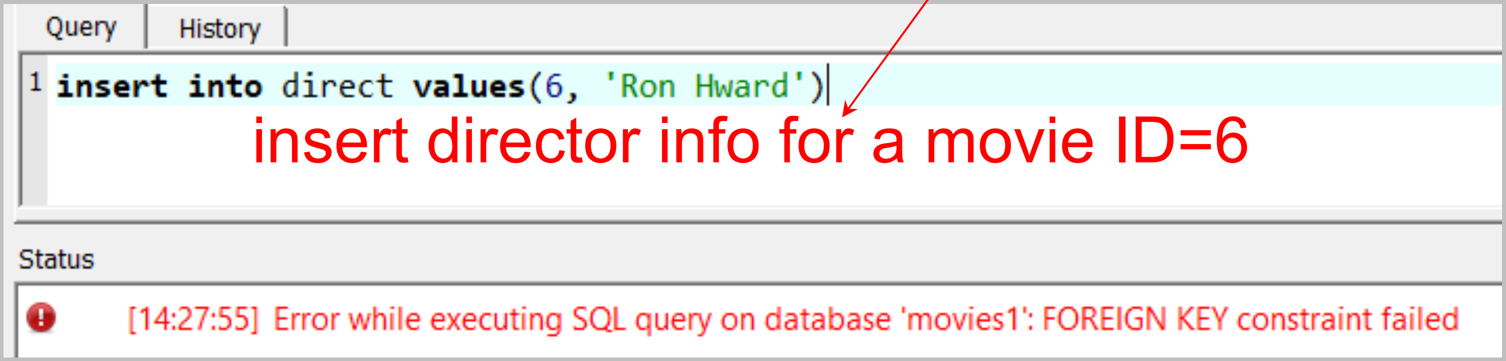
mvID	Title
1	Angels & Demons
2	Coco Avant Chanel
3	Harry Potter and the Half-Blood Prince
4	The Proposal
5	Ice Age: Dawn of the Dinosaurs



Direct

mvID	Director
1	Ron Howard
2	Anne Fontaine
3	David Yates
4	Anne Fletcher
5	Carlos Saldanha
5	Mike Thurmeier

Referential integrity constraint violated!



```
insert into direct values(6, 'Ron Hward')
```

insert director info for a movie ID=6

Status

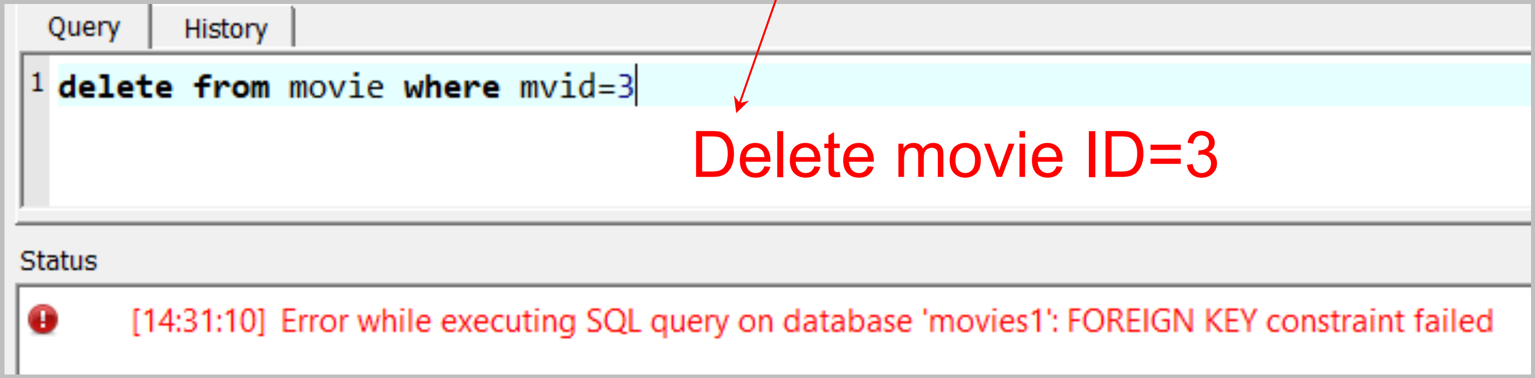
```
[14:27:55] Error while executing SQL query on database 'movies1': FOREIGN KEY constraint failed
```

Foreign keys …

- The foreign key referential integrity constraint ensures sensible deletion. A movie is deleted only when it is not referenced by any other relations.
- Example: To delete the movie “Happy Potter” from the database.

```
delete from movie where mvid=3;
```

Referential integrity constraint violated!



```
delete from movie where mvid=3
```

Delete movie ID=3

```
[14:31:10] Error while executing SQL query on database 'movies1': FOREIGN KEY constraint failed
```

Summary

- Tables and relations are used interchangeably, even though relations have a much stricter mathematical definition.
- The primary key and foreign key are the most important *integrity constraints* in the relational model. They help make sure that the data kept in databases are sensible.
  - Other integrity constraints can be defined in Oracle.
- SQL as a *data definition language*.

```
  CREATE TABLE ... (
        ....
      PRIMARY KEY (...)
       FOREIGN KEY (...) REFERENCES ....
     );
```