

The Movie Database

[UML version]

Ellen Munthe-Kaas

April 2010

The following is a slightly shorter version in English of a document written in conjunction with the mandatory exercises in INF3100.

Background

The test database to be used in the mandatory exercises is a version of the Internet Movie Database (*imdb*) [1], which is a large database containing information on approximately 700000 films and 60000 TV series, 1.7 million persons related to the films, descriptions of the films etc¹. The database runs on Postgres.

History

A first version containing a subset of the films was realized in 2002² on Sybase ASE 11.9.2. Later the department migrated to Oracle. In 2007 the department decided to migrate to Postgres; in connection with this we have decided to include a more or less full version of *imdb*.

Design of the 2007 Version

Films and TV Series

The 2007 version of the database contains ordinary cinema films as well as TV films and series etc. *FilmItem* contains all kinds of "films", from ordinary cinema films to episodes in TV series. For each collection of episodes that make up a series, *FilmItem* in addition contains an item that represents the series as such, and which can be used to find the title of the series.

¹ April 2007.

² The primary force behind the 2002 database was Igor Rafienko. David Ranvig converted *imdb* files to SQL insert sentences. Rune Aske assisted in testing the *imdb* mirror.

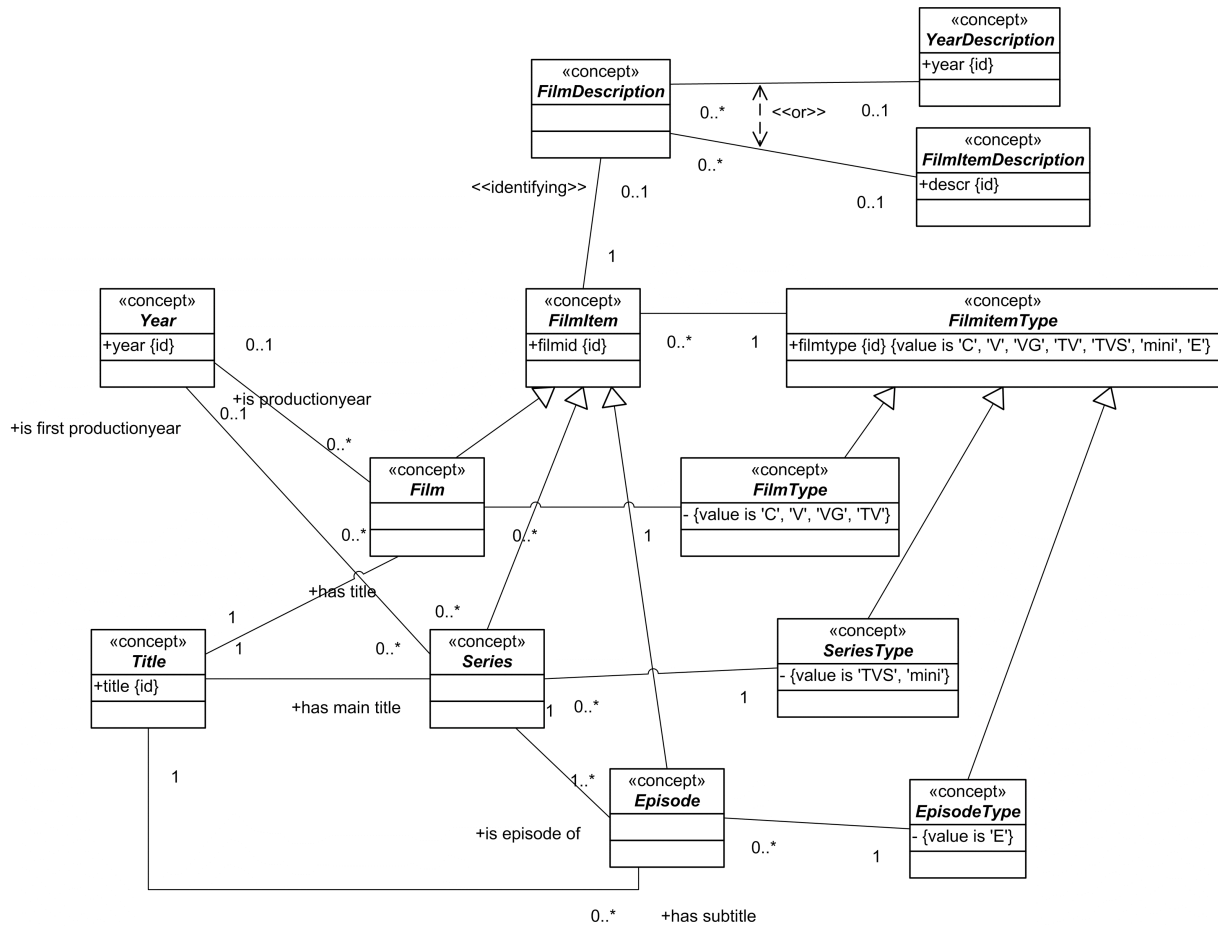


Figure 2: FilmItem and its subconcepts

FilmItem types:

- C ordinary cinema film
- V video film
- VG video game
- TV TV film
- TVS TV series
- mini mini TV series
- E episode in a TV series or a mini TV series. The corresponding occurrence in Series has type TVS or mini.

filmdescription contains some additional information for some of the filmitems.

```

create table filmitem (
    filmid int primary key,
    filmtyp varchar(4) not null
);

```

```

create table film (
    filmid int primary key references filmitem (filmid),
    title text not null,
    prodyear int
);

```

```

create index filmtitleindex on film (title);
create index filmyearindex on film (prodyear);

```

```
create table filmdescription (  
  filmid int primary key references filmitem (filmid),  
  year text,  
  filmdescr text,  
  check (year is not null or filmdescr is not null)  
);  
  
create table series (  
  seriesid int primary key references filmitem (filmid),  
  maintitle text not null,  
  firstprodyear int  
);  
create index seriesmaintitleindex on series (maintitle);  
  
create table episode (  
  episodeid int primary key references filmitem (filmid),  
  seriesid int not null references filmitem (filmid),  
  subtitle text not null,  
  foreign key (seriesid) references series (seriesid)  
);
```

Additional Film Information

For some films alternative titles (AlternativeFilmTitle), production country (FilmCountry), and language (FilmLanguage) exist. Some films contain some additional information about the language (FilmLanguageInfo). A film can belong to one or more genres (FilmGenre).

RunningTime contains the length of the film when run in different countries, potentially with some additional information (RunningTimeInfo). imdb allows the audience to vote (points between 1 and 10) for films, this is included in FilmRating. Distribution contains how the votes distribute over the points given. Some films are assigned a rank based on this.

Distribution is a 10-character string.

1. character: How many voted 1 point (lowest score)
2. "- 2 points
- ...
- 10 "- 10 points (best score)

The characters are interpreted as follows:

- "." no votes
- "0" 1-9%
- "1" 10-19%
- ...
- "9" 90-99%
- "*" 100%

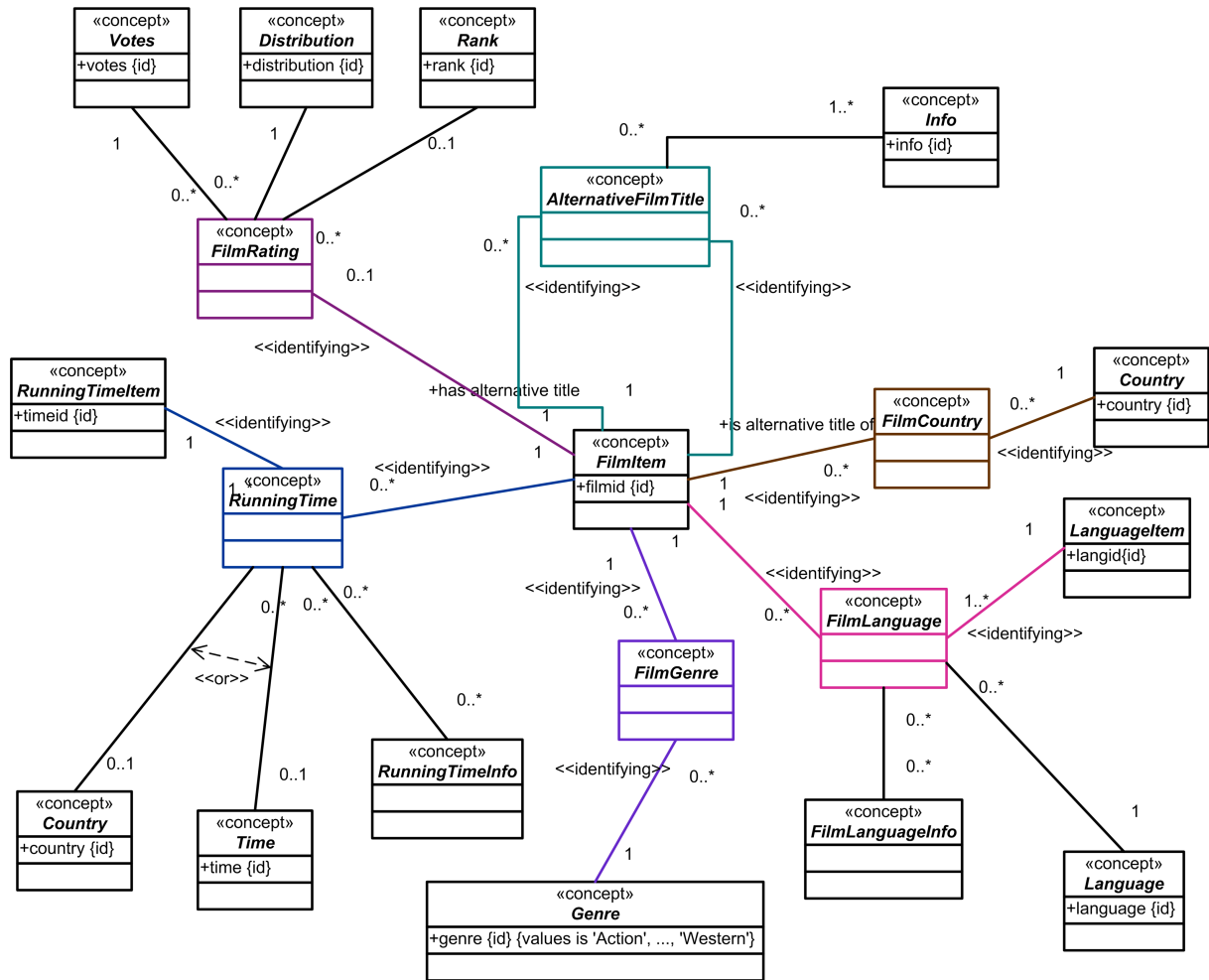


Figure 3: Additional film information

```

create table alternativefilmtitle (
  filmid int references filmitem (filmid),
  akaid int references filmitem (filmid),
  info text not null
);

create index alternativefilmtitlefilmidindex
  on alternativefilmtitle (filmid);
create index alternativefilmtitleakaidindex
  on alternativefilmtitle (akaid);

create table filmcountry (
  filmid int references filmitem (filmid),
  country text,
  primary key (filmid, country)
);

create index filmcountryfilmidindex on filmcountry (filmid);

create table country (
  country text primary key
);

```

```
create table filmlanguage (  
    filmid int references filmlitem (filmid),  
    langid int,  
    language text not null,  
    primary key (filmid, langid)  
);  
  
create index filmlanguagefilmidindex on filmlanguage (filmid);  
  
create table filmlanguageinfo (  
    filmid int not null,  
    langid int not null,  
    info text not null,  
    foreign key (filmid, langid) references filmlanguage (filmid, langid)  
);  
  
create index filmlanguageinfofilmidlangidindex  
    on filmlanguageinfo (filmid, langid);  
  
create table language (  
    language text primary key  
);  
  
create table filmgenre (  
    filmid int references filmlitem (filmid),  
    genre text,  
    primary key (filmid, genre)  
);  
  
create index filmgenrefilmidindex on filmgenre (filmid);  
create index filmgenregenreindex on filmgenre (genre);  
  
create table genre (  
    genre text primary key  
);  
  
create table runningtime (  
    filmid int references filmlitem (filmid),  
    timeid int,  
    time text,  
    country text,  
    primary key (filmid, timeid),  
    check (time is not null or country is not null)  
);  
  
create index runningtimefilmidindex on runningtime (filmid);  
  
create table runningtimeinfo (  
    filmid int not null,  
    timeid int not null,  
    info text not null,  
    foreign key (filmid, timeid) references runningtime (filmid, timeid)  
);  
  
create index runningtimeinfotimeidindex on runningtimeinfo (filmid,  
timeid);
```

```
create table filmrating (  
  filmid int primary key references filmitem (filmid),  
  votes int not null,  
  distribution char(10) not null,  
  rank float(3)  
);
```

Persons

All persons have a family name, and mostly also a first name. Gender is only present for some persons. hasFirstName includes first and middle names. In BiographyItem are placed fairly long texts under a number of different codes, e.g.:

- RN: Real name
- TR: Trade
- DB: Date of birth
- DD: Date of death
- NK: Nickname
- BG: Background
- BY: Biographer
- SP: Spouse
- HT: Height
- OW: Other work
- CV: Curriculum vitae
- QU: Quotations

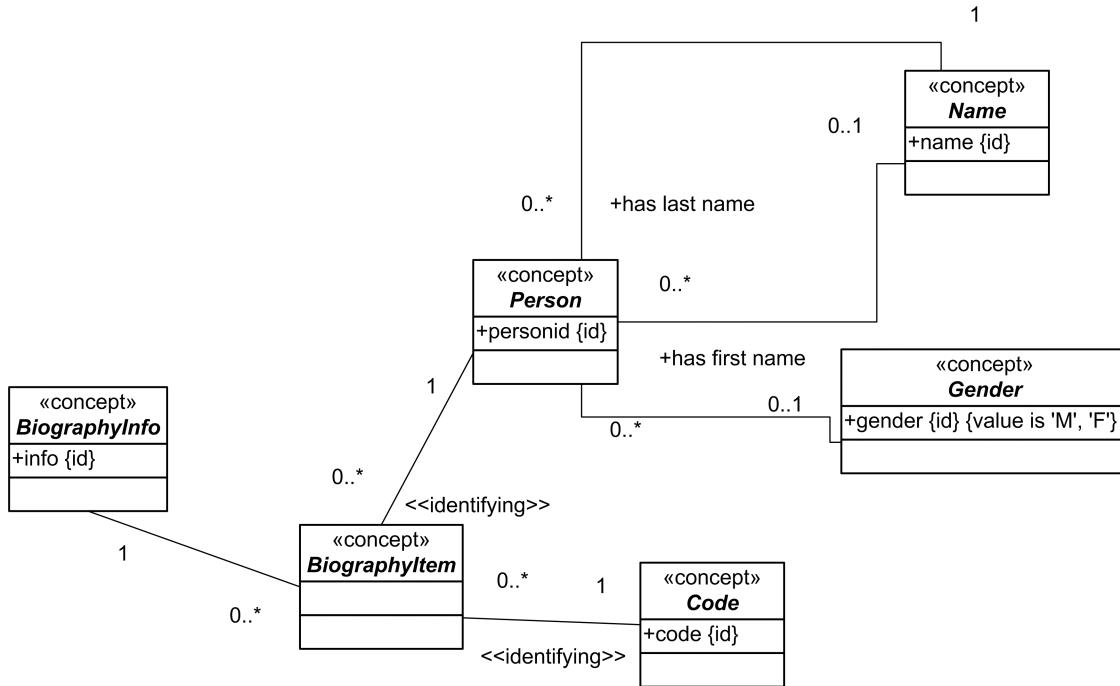


Figure 4: Person

```
create table person (  
  personid int primary key,  
  lastname text not null,  
  firstname text not null,  
  gender char(1),  
  check (gender = 'M' or gender = 'F');  
);  
  
create index personlastnameindex on person (lastname);  
  
create table biographyitem (  
  personid int references person (personid),  
  code char(2),  
  description text not null,  
  primary key (personid, code)  
);
```

Filmparticipation

Persons can participate in films as cast, composer, costume designer, director, editor, producer, writer. Film participation is contained in FilmParticipation. Some additional information may be found in FilmParticipationInfo.

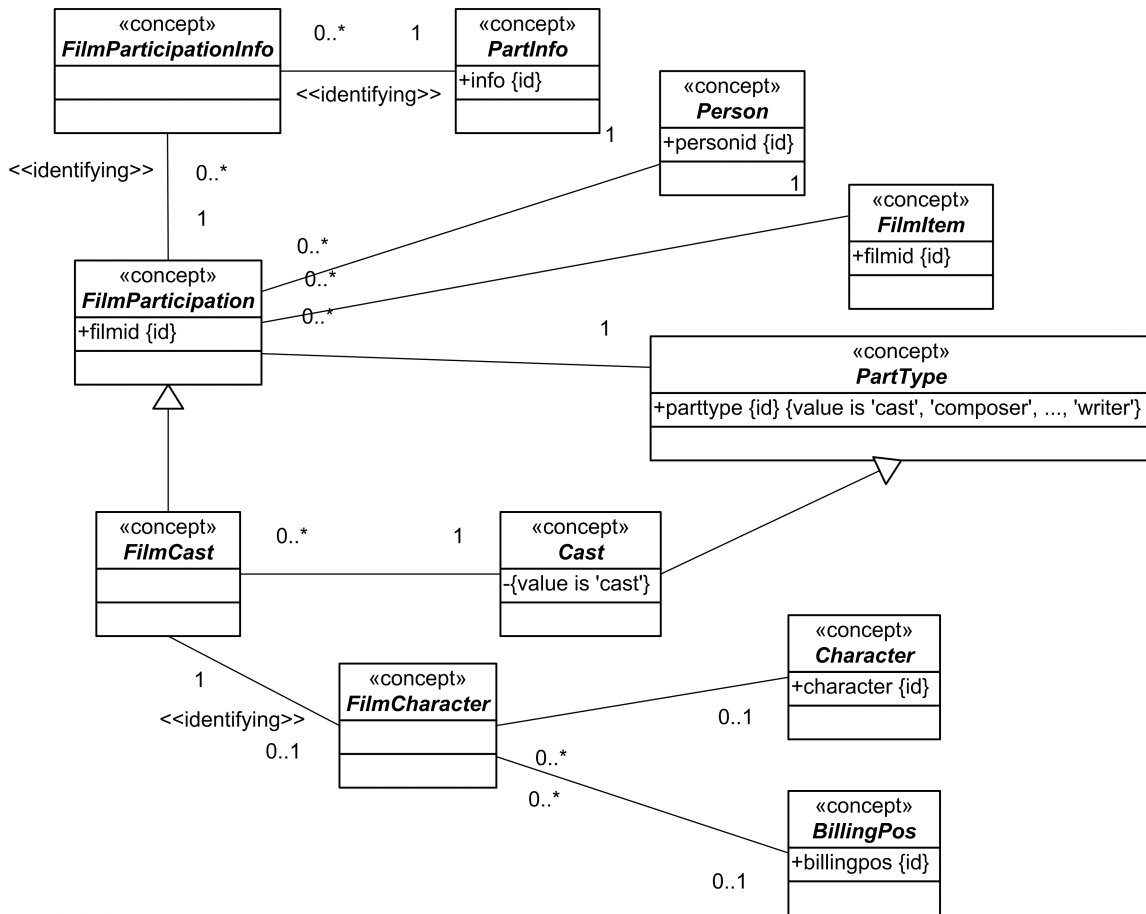


Figure 5: Filmparticipation

```
create table filmparticipation (  
  partid int primary key,  
  personid int not null references person (personid),  
  filmid int not null references filmitem (filmid),  
  parttype text not null  
);  
  
create index filmparticipationpersonidindex  
  on filmparticipation (personid);  
create index filmparticipationfilmidindex  
  on filmparticipation (filmid);  
  
create table filmcharacter (  
  partid int primary key references filmparticipation (partid),  
  filmcharacter text,  
  billingpos int,  
  check (filmcharacter is not null or billingpos is not null)  
);  
create table filmparticipationinfo (  
  partid int not null references filmparticipation (partid),  
  info text not null  
);  
  
create index filmparticipationinfopartidindex  
  on filmparticipationinfo (partid);
```

Examples

Luc Besson's participation in the film The Fifth Element:

```
username=> select x.partid, x.personid, x.filmid, x.parttype  
username-> from filmparticipation x, person p, film f  
username-> where  
username->   p.lastname = 'Besson' and  
brukeranvn->   p.firstname = 'Luc' and  
username->   f.title = 'Fifth Element, The' and  
username->   x.personid = p.personid and  
username->   x.filmid = f.filmid;  
  partid | personid | filmid | parttype  
-----+-----+-----+-----  
  781285 |    89222 | 237127 | director  
 1009544 |    89222 | 237127 | writer  
 1009560 |    89222 | 237127 | writer  
 1009576 |    89222 | 665467 | writer  
 1009592 |    89222 | 665467 | writer  
(5 rows)
```

username=>

Milla Jovovich's roles in The Fifth Element:

```
username=> select filmcharacter  
username-> from filmcharacter  
username-> where partid = 19580594 or partid = 19580610;  
  filmcharacter  
-----  
  Leeloo  
  Leeloo  
(2 rows)
```

username=>

Additional information about these roles:

```
username=> select partid, info
username-> from filmparticipationinfo
username-> where partid = 19580594 or partid = 19580610;
  partid | info
-----+-----
 19580610 | voice
(1 row)

username=>
```

The reason why there are two filmids for The Fifth Element, is that there is a VG version of the film:

```
username=> select f.filmid, title, prodyear, filmtype
username-> from film f, filmitem i
username-> where
username->   f.title = 'Fifth Element, The' and
username->   f.filmid = i.filmid;
  filmid |          title          | prodyear | filmtype
-----+-----+-----+-----
 237127 | Fifth Element, The |    1997 | C
 665467 | Fifth Element, The |    1998 | VG
(2 rows)

username=>
```

All roles played by Milla Jovovich:

```
username=> select f.title, c.filmcharacter
username-> from filmparticipation x, person p, film f, filmcharacter c
username-> where
username->   x.parttype = 'cast' and
username->   p.lastname = 'Jovovich' and
username->   p.firstname = 'Milla' and
username->   x.personid = p.personid and
username->   x.filmid = f.filmid and
username->   x.partid = c.partid;
          title          | filmcharacter
-----+-----
.45                      | Kate
AFI's 100 Years... 100 Cheers: America's Most Inspiring Movies | Herself
Cannes: Through the Eyes of the Hunter                       | Herself
Chaplin                                                          | Mildred Harris
Claim, The                                                      | Lucia
Corporate Malfeasance                                         | Herself
Dazed and Confused                                             | Michelle Burroughs
Dummy                                                          | Fangora
Fifth Element, The                                            | Leeloo
Fifth Element, The                                            | Leeloo
Game Babes                                                     | Herself
Game Over: 'Resident Evil' Reanimated                        | Herself
He Got Game                                                    | Dakota Burns
House on Turk Street, The                                     | Erin
Kuffs                                                          | Maya Carlton
Making and Meaning of 'We Are Family', The                  | Herself
Messenger: The Story of Joan of Arc, The                     | Joan of Arc
Million Dollar Hotel, The                                     | Eloise
Night Train to Kathmandu, The                                 | Lily McLeod
Playing Dead: 'Resident Evil' from Game to Screen           | Herself
Resident Evil                                                 | Alice
Resident Evil: Apocalypse                                     | Alice
```

```
Resident Evil: Extinction | Alice
Return to the Blue Lagoon | Lilli
Star Element, The | Herself
Starz on the Set: Ultraviolet | Herself
Teen Vid II | Herself
Trailer for a Remake of Gore Vidal's Caligula | Druscilla
Two Moon Junction | Samantha Delongpre
Ultraviolet | Violet
VH1/Vogue Fashion Awards | Herself
You Stupid Man | Nadine
Zoolander | Katinka
(33 rows)
```

username=>

These are only her cinema films. Milla Jovovich has played in TV series too:

```
username=> select s.maintitle, e.subtitle, c.filmcharacter
username-> from filmparticipation x, person p,
username-> episode e, filmcharacter c, series s
username-> where
username-> x.parttype = 'cast' and
username-> p.lastname = 'Jovovich' and
username-> p.firstname = 'Milla' and
username-> x.personid = p.personid and
username-> x.filmid = e.episodeid and
username-> x.partid = c.partid and
username-> e.seriesid = s.seriesid;
```

maintitle	subtitle	filmcharacter
4Pop	Nuo surkeat Hollywood-pelit (#3.6)	Herself
Grand journal de Canal+, Le	(2005-05-20)	Herself
Harald Schmidt Show, Die	(2002-03-19)	Herself
HBO First Look	The Messenger: The Story of Joan of Arc	Herself
HypaSpace	(#5.39)	Herself
HypaSpace	(#5.40)	Herself
HypaSpace	(#5.42)	Herself
HypaSpace	(#5.44)	Herself
HypaSpace	(#5.45)	Herself
King of the Hill	Get Your Freak Off (#7.1)	Serena
Late Late Show with Craig Ferguson, The	(#2.104)	Herself
Married with Children	Fair Exchange (#4.6)	Yvette
Paradise	Childhood's End (#1.8)	Katie
Parker Lewis Can't Lose	Pilot (#1.1)	Robin Fecknowitz
Tout le monde en parle	(2002-03-23)	Herself
V Graham Norton	(#1.47)	Herself

username=>

References

[1] The Internet Movie Database, <http://www.imdb.com/>