



PROIECT SISTEME DE GESTIUNE A BAZELOR DE DATE



Dudau Claudia Maria

1. Prezentare pe scurt a bazei de date

O baza de date ce conține informații despre seriale, producători, sezoane, episoade, actori și personaje, în scopul manipulării acestora.

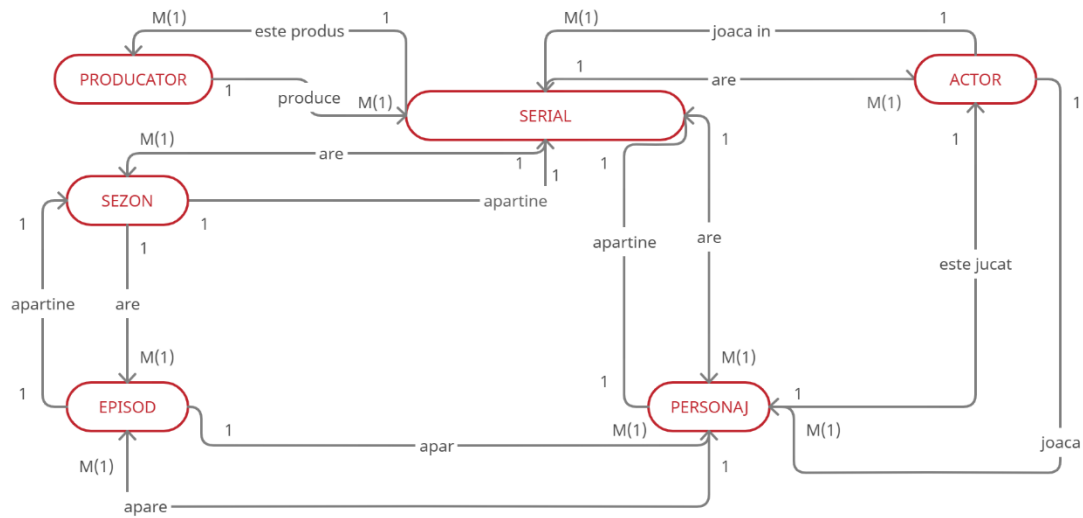
Ca orice baza de date, scopul său principal este acela de a oferi o structură logică a componentelor și a relațiilor dintre acestea pentru ca informațiile să poată fi accesate cu ușurință.

Aceasta este utilă în special site-urilor ce se ocupă cu oferirea de informații legate de diverse seriale (cum ar fi imdb), dar și pentru aplicații care se ocupă cu gestionarea datelor de acest fel (adăugare, modificare, ștergere).

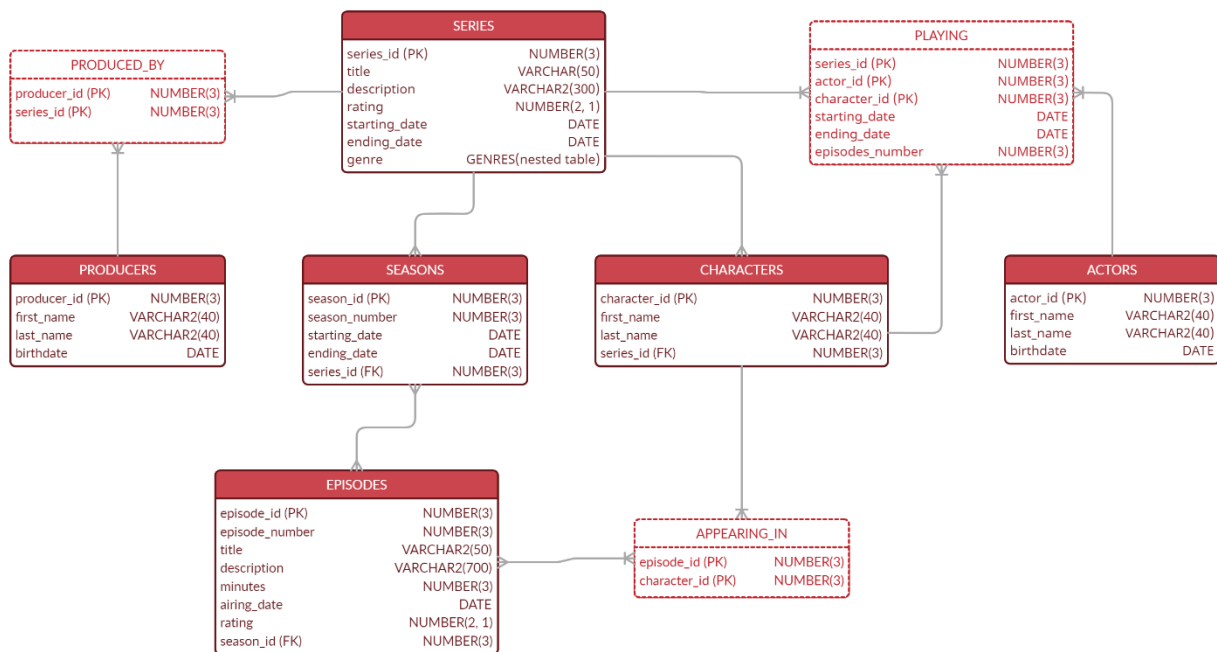
În acest sens, baza de date este concepută cu 6 tabele independente (producers, series, seasons, episodes, actors, characters) și 3 tabele asociative pentru a rezolva legăturile de tip many to many (produced_by, appearing_in, playing). Din punct de vedere al relațiilor, sunt implementate următoarele reguli:

- un producător poate produce unul sau mai multe seriale, iar un serial poate fi produs de unul sau mai mulți producători;
- un actor joacă în unul sau mai multe seriale, un serial având unul sau mai mulți actori;
- un serial are unul sau mai multe sezoane, iar un sezon aparține unui singur serial;
- un sezon are unul sau mai multe episoade, iar un episod aparține unui singur sezon;
- un actor joacă unul sau mai multe personaje, iar un personaj este jucat de un singur actor;
- un personaj apare în unul sau mai multe episoade, în timp ce într-un episod pot apărea unul sau mai multe personaje.

2. Diagrama E/R



3. Diagrama conceptuala



4. Implementarea in Oracle a diagramei conceptuale

1) Tabela series

Am ales să stochez categoriile unui serial sub forma de tablou imbricat.

```
CREATE OR REPLACE TYPE genres IS TABLE OF VARCHAR2(20);
/
```

```
CREATE TABLE series
```

```
(series_id NUMBER(3) PRIMARY KEY,
 title VARCHAR2(50) NOT NULL,
 description VARCHAR(300),
 rating NUMBER(2,1),
 starting_date DATE NOT NULL,
 ending_date DATE,
 genre genres,
 CONSTRAINT date_series CHECK (starting_date <= ending_date))
```

```
NESTED TABLE genre STORE AS genre;
```

Name	Null?	Type
-----	-----	-----
SERIES_ID	NOT NULL	NUMBER(3)
TITLE	NOT NULL	VARCHAR2(50)
DESCRIPTION		VARCHAR2(300)
RATING		NUMBER(2,1)
STARTING_DATE	NOT NULL	DATE
ENDING_DATE		DATE
GENRE		GENRES

2) Tabela producers

```
CREATE TABLE producers
```

```
(producer_id NUMBER(3) PRIMARY KEY,
 first_name VARCHAR2(40) NOT NULL,
 last_name VARCHAR2(40) NOT NULL,
 birthdate DATE);
```

Name	Null?	Type
-----	-----	-----
PRODUCER_ID	NOT NULL	NUMBER(3)
FIRST_NAME	NOT NULL	VARCHAR2(40)
LAST_NAME	NOT NULL	VARCHAR2(40)
BIRTHDATE		DATE

3) Tabela produced_by

```
CREATE TABLE produced_by
  (producer_id NUMBER(3) REFERENCES producers(producer_id) ON DELETE
  CASCADE,
  series_id NUMBER(3) REFERENCES series(series_id) ON DELETE CASCADE,
  PRIMARY KEY(producer_id, series_id));
```

Name	Null?	Type
-----	-----	-----
PRODUCER_ID	NOT NULL	NUMBER(3)
SERIES_ID	NOT NULL	NUMBER(3)

4) Tabela seasons

```
CREATE TABLE seasons
  (season_id NUMBER(3) PRIMARY KEY,
  season_number NUMBER(2) NOT NULL,
  starting_date DATE,
  ending_date DATE,
  series_id NUMBER(3) REFERENCES series(series_id) ON DELETE CASCADE);
```

Name	Null?	Type
-----	-----	-----
SEASON_ID	NOT NULL	NUMBER(3)
SEASON_NUMBER	NOT NULL	NUMBER(2)
STARTING_DATE		DATE
ENDING_DATE		DATE
SERIES_ID		NUMBER(3)

5) Tabela episodes

```
CREATE TABLE episodes
(episode_id NUMBER(3) PRIMARY KEY,
episode_number NUMBER(3) NOT NULL,
title VARCHAR2(50) NOT NULL,
description VARCHAR2 (700),
minutes NUMBER(3),
airing_date DATE,
rating NUMBER(2,1),
season_id NUMBER(3) REFERENCES seasons(season_id) ON DELETE CASCADE);
```

Name	Null?	Type
EPISODE_ID	NOT NULL	NUMBER (3)
EPISODE_NUMBER	NOT NULL	NUMBER (3)
TITLE	NOT NULL	VARCHAR2 (50)
DESCRIPTION		VARCHAR2 (700)
MINUTES		NUMBER (3)
AIRING_DATE		DATE
RATING		NUMBER (2, 1)
SEASON_ID		NUMBER (3)

6) Tabela actors

```
CREATE TABLE actors
(actor_id NUMBER(3) PRIMARY KEY,
first_name VARCHAR2(40) NOT NULL,
last_name VARCHAR2(40) NOT NULL,
birth_date DATE);
```

Name	Null?	Type
ACTOR_ID	NOT NULL	NUMBER (3)
FIRST_NAME	NOT NULL	VARCHAR2 (40)
LAST_NAME	NOT NULL	VARCHAR2 (40)
BIRTH_DATE		DATE

7) Tabela characters

```
CREATE TABLE characters
(character_id NUMBER(3) PRIMARY KEY,
first_name VARCHAR2(40) NOT NULL,
last_name VARCHAR2(40),
series_id NUMBER(3) REFERENCES series(series_id) ON DELETE CASCADE);
```

Name	Null?	Type
-----	-----	-----
CHARACTER_ID	NOT NULL	NUMBER(3)
FIRST_NAME	NOT NULL	VARCHAR2(40)
LAST_NAME		VARCHAR2(40)
SERIES_ID		NUMBER(3)

8) Tabela playing

```
CREATE TABLE playing
(series_id NUMBER(3) REFERENCES series(series_id) ON DELETE CASCADE,
actor_id NUMBER(3) REFERENCES actors(actor_id) ON DELETE CASCADE,
character_id NUMBER(3) REFERENCES characters(character_id) ON DELETE
CASCADE,
starting_date DATE,
ending_date DATE,
episodes_number NUMBER(3),
PRIMARY KEY(series_id, character_id, actor_id),
CONSTRAINT date_playing CHECK (starting_date <= ending_date));
```

Name	Null?	Type
-----	-----	-----
SERIES_ID	NOT NULL	NUMBER(3)
ACTOR_ID	NOT NULL	NUMBER(3)
CHARACTER_ID	NOT NULL	NUMBER(3)
STARTING_DATE		DATE
ENDING_DATE		DATE
EPISODES_NUMBER		NUMBER(3)

9) Tabela appearing_in

```
CREATE TABLE appearing_in
  (episode_id NUMBER(3) REFERENCES episodes(episode_id) ON DELETE CASCADE,
   character_id NUMBER(3) REFERENCES characters(character_id) ON DELETE
  CASCADE,
   PRIMARY KEY(episode_id, character_id));
```

Name	Null?	Type
-----	-----	-----
EPISODE_ID	NOT NULL	NUMBER(3)
CHARACTER_ID	NOT NULL	NUMBER(3)

5. Adăugarea de informații în tabele

1) Tabela series

```
INSERT INTO series
VALUES (1, 'Supernatural',
       'Two brothers follow their father's footsteps as hunters, fighting evil supernatural beings of
       many kinds, including monsters, demons and gods that roam the earth.',
       8.4, TO_DATE('13-Sep-2005', 'DD MONTH YYYY'), TO_DATE('19-Nov-2020', 'DD
       MONTH YYYY'), genres('Drama', 'Fantasy', 'Horror'));
```

```
INSERT INTO series
VALUES (2, 'Gossip Girl',
       'Privileged teens living on the Upper East Side of New York can hide no secret from the
       ruthless blogger who is always watching.',
       7.4, TO_DATE('19-Sep-2007', 'DD MONTH YYYY'), TO_DATE('17-Dec-2012', 'DD
       MONTH YYYY'), genres('Drama', 'Romance'));
```

```
INSERT INTO series
VALUES (3, 'The Originals',
       'A family of power-hungry thousand-year-old vampires look to take back the city that they
       built and dominate all those who have done them wrong.',
       8.2, TO_DATE('3-Oct-2013', 'DD MONTH YYYY'), TO_DATE('1-Aug-2018', 'DD
       MONTH YYYY'), genres('Drama', 'Fantasy', 'Horror'));
```


INSERT INTO series

VALUES (4, 'Arrow',

'Spoiled billionaire playboy Oliver Queen is missing and presumed dead when his yacht is lost at sea. He returns five years later a changed man, determined to clean up the city as a hooded vigilante armed with a bow.',

7.5, TO_DATE('10-Oct-2012', 'DD MONTH YYYY'), TO_DATE('28-Jan-2020', 'DD MONTH YYYY'), genres('Action', 'Adventure', 'Crime'));

INSERT INTO series

VALUES (5, 'The 100',

'Set ninety-seven years after a nuclear war has destroyed civilization, when a spaceship housing humanity's lone survivors sends one hundred juvenile delinquents back to Earth, in hopes of possibly re-populating the planet.',

7.6, TO_DATE('19-Mar-2014', 'DD MONTH YYYY'), TO_DATE('30-Sep-2020', 'DD MONTH YYYY'), genres('Drama', 'Mystery', 'Sci-Fi'));

INSERT INTO series

VALUES (6, 'Lucifer',

'Lucifer Morningstar has decided he's had enough of being the dutiful servant in Hell and decides to spend some time on Earth to better understand humanity. He settles in Los Angeles - the City of Angels.',

8.2, TO_DATE('25-Jan-2016', 'DD MONTH YYYY'), NULL, genres('Crime', 'Drama', 'Fantasy'));

INSERT INTO series

VALUES (7, 'The Magicians',

'After being recruited to a secretive academy, a group of students discover that the magic they read about as children is very real-and more dangerous than they ever imagined.',

7.6, TO_DATE('26-Dec-2015', 'DD MONTH YYYY'), TO_DATE('1-Apr-2020', 'DD MONTH YYYY'), genres('Drama', 'Fantasy', 'Mystery'));

SERIES_ID	TITLE	DESCRIPTION
1	Supernatural	Two brothers follow their father's footsteps as
2	Gossip Girl	Privileged teens living on the Upper East Side o
3	The Originals	A family of power-hungry thousand-year-old vampi
4	Arrow	Spoiled billionaire playboy Oliver Queen is miss
5	The 100	Set ninety-seven years after a nuclear war has d
6	Lucifer	Lucifer Morningstar has decided he's had enough
7	The Magicians	After being recruited to a secretive academy, a

2) Tabela producers

```
INSERT INTO producers
VALUES (1, 'Eric', 'Kripke', '24-APR-1974');
```

```
INSERT INTO producers
VALUES (2, 'Stephanie', 'Savage', TO_DATE('1969', 'YYYY'));
```

```
INSERT INTO producers
VALUES (3, 'Josh', 'Schwartz', '06-AUG-1976');
```

```
INSERT INTO producers
VALUES (4, 'Julie', 'Plec', '26-MAY-1972');
```

```
INSERT INTO producers
VALUES (5, 'Greg', 'Berlanti', '24-MAY-1972');
```

```
INSERT INTO producers
VALUES (6, 'Marc', 'Guggenheim', '24-SEP-1970');
```

```
INSERT INTO producers
VALUES (7, 'Andrew', 'Kreisberg', '23-APR-1971');
```

```
INSERT INTO producers
VALUES (8, 'Jason', 'Rothenberg', null);
```

```
INSERT INTO producers
VALUES (9, 'Tom', 'Kapos', TO_DATE('1969', 'YYYY'));
```

	PRODUCER_ID	FIRST_NAME	LAST_NAME	BIRTHDATE
1	1	Eric	Kripke	24-APR-74
2	2	Stephanie	Savage	01-DEC-69
3	3	Josh	Schwartz	06-AUG-76
4	4	Julie	Plec	26-MAY-72
5	5	Greg	Berlanti	24-MAY-72
6	6	Marc	Guggenheim	24-SEP-70
7	7	Andrew	Kreisberg	23-APR-71
8	8	Jason	Rothenberg	(null)
9	9	Tom	Kapos	01-DEC-69

3) Tabela produced_by

```
INSERT INTO produced_by  
VALUES (1, 1);
```

```
INSERT INTO produced_by  
VALUES (2, 2);
```

```
INSERT INTO produced_by  
VALUES (3, 2);
```

```
INSERT INTO produced_by  
VALUES (4, 3);
```

```
INSERT INTO produced_by  
VALUES (5, 4);
```

```
INSERT INTO produced_by  
VALUES (6, 4);
```

```
INSERT INTO produced_by  
VALUES (7, 4);
```

```
INSERT INTO produced_by  
VALUES (8, 5);
```

```
INSERT INTO produced_by  
VALUES (9, 6);
```

	PRODUCER_ID	SERIES_ID
1	1	1
2	2	2
3	3	2
4	4	3
5	5	4
6	6	4
7	7	4
8	8	5
9	9	6

4) Tabela seasons

```
INSERT INTO seasons
VALUES (1, 5, TO_DATE('10-Sep-2009', 'DD MONTH YYYY'), TO_DATE('13-May-2010', 'DD MONTH YYYY'), 1);
```

```
INSERT INTO seasons
VALUES (2, 15, TO_DATE('10-Oct-2019', 'DD MONTH YYYY'), TO_DATE('19-Nov-2020', 'DD MONTH YYYY'), 1);
```

```
INSERT INTO seasons
VALUES (3, 1, TO_DATE('19-Sep-2007', 'DD MONTH YYYY'), TO_DATE('19-May-2008', 'DD MONTH YYYY'), 2);
```

```
INSERT INTO seasons
VALUES (4, 6, TO_DATE('8-Oct-2011', 'DD MONTH YYYY'), TO_DATE('17-Dec-2012', 'DD MONTH YYYY'), 2);
```

```
INSERT INTO seasons
VALUES (5, 1, TO_DATE('3-Oct-2013', 'DD MONTH YYYY'), TO_DATE('13-May-2014', 'DD MONTH YYYY'), 3);
```

```
INSERT INTO seasons
VALUES (6, 3, TO_DATE('8-Oct-2015', 'DD MONTH YYYY'), TO_DATE('20-May-2016', 'DD MONTH YYYY'), 3);
```

```
INSERT INTO seasons
VALUES (7, 2, TO_DATE('9-Oct-2013', 'DD MONTH YYYY'), TO_DATE('14-May-2014', 'DD MONTH YYYY'), 4);
```

```
INSERT INTO seasons
VALUES (8, 3, TO_DATE('8-Oct-2014', 'DD MONTH YYYY'), TO_DATE('13-May-2015', 'DD MONTH YYYY'), 4);
```

```
INSERT INTO seasons
VALUES (9, 1, TO_DATE('19-Mar-2014', 'DD MONTH YYYY'), TO_DATE('11-Jun-2014', 'DD MONTH YYYY'), 5);
```

```
INSERT INTO seasons
VALUES (10, 5, TO_DATE('24-Apr-2018', 'DD MONTH YYYY'), TO_DATE('7-Aug-2018', 'DD MONTH YYYY'), 5);
```

INSERT INTO seasons

VALUES (11, 1, TO_DATE('25-Jan-2016', 'DD MONTH YYYY'), TO_DATE('25-Apr-2016', 'DD MONTH YYYY'), 6);

INSERT INTO seasons

VALUES (12, 4, TO_DATE('8-May-2019', 'DD MONTH YYYY'), TO_DATE('8-May-2019', 'DD MONTH YYYY'), 6);

	SEASON_ID	SEASON_NUMBER	STARTING_DATE	ENDING_DATE	SERIES_ID
1	1	5	10-SEP-09	13-MAY-10	1
2	2	15	10-OCT-19	19-NOV-20	1
3	3	1	19-SEP-07	19-MAY-08	2
4	4	6	08-OCT-11	17-DEC-12	2
5	5	1	03-OCT-13	13-MAY-14	3
6	6	3	08-OCT-15	20-MAY-16	3
7	7	2	09-OCT-13	14-MAY-14	4
8	8	3	08-OCT-14	13-MAY-15	4
9	9	1	19-MAR-14	11-JUN-14	5
10	10	5	24-APR-18	07-AUG-18	5
11	11	1	25-JAN-16	25-APR-16	6
12	12	4	08-MAY-19	08-MAY-19	6

5) Tabela episodes

INSERT INTO episodes

VALUES (1, 22, 'Swan Song', 'With the Apocalypse looming, Sam and Dean realize they are out of options and make heart-breaking decisions that will change their lives forever.', 43, TO_DATE('13-May-2010', 'DD MONTH YYYY'), 9.7, 1);

INSERT INTO episodes

VALUES (2, 8, 'Changing Channels', 'Sam and Dean catch up with the Trickster, who sends them through a dizzying montage of TV show parodies, inviting them to play along with their "roles" or be stuck in "TV Land" forever. But once Castiel shows up, the boys get an idea as to what the Trickster might be hiding and eventually come up with a surprising answer.', 43, TO_DATE('5-Nov-2009', 'DD MONTH YYYY'), 9.7, 1);

INSERT INTO episodes

VALUES (3, 19, 'Inherit the Earth', 'Everything is on the line as the battle against God continues; a familiar face returns to join the fight.', 43, TO_DATE('12-Nov-2020', 'DD MONTH YYYY'), 8.2, 2);

INSERT INTO episodes

VALUES (4, 10, 'New York, I Love You XOXO', 'In a fashionable farewell to remember, our favorite Upper East Siders join forces for one last soiree; The identity of Gossip Girl is finally revealed.',

41, TO_DATE('17-Dec-2012', 'DD MONTH YYYY'), 9.1, 4);

INSERT INTO episodes

VALUES (5, 7, 'Victor/Victrola', 'Serena and Dan finally acknowledge they are crazy about each other; Jenny discovers a secret about her parents; Blair is once again devastated by Nate's actions.',

42, TO_DATE('7-Nov-2007', 'DD MONTH YYYY'), 8.2, 3);

INSERT INTO episodes

VALUES (6, 22, 'From a Cradle to a Grave', 'As the baby's due date draws near, Klaus and Elijah embark on a search for Hayley, while Hayley determines to do whatever it takes to keep her unborn baby safe and away from the witches. Francesca takes a meeting with Oliver and Jackson to determine the future of the werewolves in New Orleans. In the aftermath of a surprising attack on Marcel and his vampires at the compound, Davina and Cami join resources to take down Klaus. Finally, in a desperate move to protect those most important to him, Klaus makes a heartbreaking decision.',

42, TO_DATE('13-May-2014', 'DD MONTH YYYY'), 9.6, 5);

INSERT INTO episodes

VALUES (7, 22, 'The Bloody Crown', 'After months of thwarting off dangerous threats and deadly attacks, the Mikaelson siblings finally come face to face with the one person that could lead to their ultimate demise. With the stakes higher than ever and the compound overrun by an army of his oldest sworn enemies, Klaus is put on trial for centuries of atrocities he's committed. Meanwhile, Marcel, who has been spiraling out of control following an act of betrayal by those closest to him, is stunned by the unexpected arrival of someone from his past. Finally, Elijah, Freya and Kol frantically search for a way to save their family before it's too late. Hayley also appears.',

42, TO_DATE('20-May-2016', 'DD MONTH YYYY'), 9.6, 6);

INSERT INTO episodes

VALUES (8, 14, 'A Streetcar Named Desire', 'The unexpected arrival of Stefan Salvatore may be the key to helping Freya rescue Klaus and Elijah from a magical trap.',

42, TO_DATE('26-Feb-2016', 'DD MONTH YYYY'), 9.4, 6);

INSERT INTO episodes

VALUES (9, 9, 'The Climb', 'The League of Assassins give Oliver 48 hours to find Sara's killer, or Starling City citizens will die. Oliver then has an epic confrontation with Ra's al Ghul.',

42, TO_DATE('10-Dec-2014', 'DD MONTH YYYY'), 9.6, 8);

INSERT INTO episodes

VALUES (10, 23, 'Unthinkable', 'As Oliver's face off with Slade escalates, his resolve to the no-kill rule is tested. Especially as Slade targets the woman Oliver loves.',

44, TO_DATE('14-May-2014', 'DD MONTH YYYY'), 9.5, 7);

INSERT INTO episodes

VALUES (11, 18, 'Deathstroke', 'After taking a ride home from Slade, Thea becomes his prisoner. Can Oliver and his friends save her? Also, someone close to Oliver is working for Slade, since his return from the Island after Tommy's death.',

42, TO_DATE('2-Apr-2014', 'DD MONTH YYYY'), 9.3, 7);

INSERT INTO episodes

VALUES (12, 13, 'Damocles: Part Two', 'Clarke and her friends must risk everything to fight one last battle for survival, only to glimpse an even darker threat to the last living valley on earth.',

42, TO_DATE('7-Aug-2018', 'DD MONTH YYYY'), 9.6, 10);

INSERT INTO episodes

VALUES (13, 12, 'Damocles: Part One', 'In part one of the fifth season finale, Octavia leads her people into war. While behind enemy lines, our heroes must overcome their differences to save Wonkru from extinction.',

42, TO_DATE('31-Jul-2018', 'DD MONTH YYYY'), 9.0, 10);

INSERT INTO episodes

VALUES (14, 12, 'We Are Grounders: Part 2', 'As the remaining members of the 100 face off against the Grounders, Jaha makes a noble sacrifice to ensure the Ark makes it to Earth.',

42, TO_DATE('11-Jul-2018', 'DD MONTH YYYY'), 8.9, 9);

INSERT INTO episodes

VALUES (15, 11, 'We Are Grounders: Part 1', 'Clarke and Finn come face to face with a new enemy after Lincoln rescues them from Anya, while Bellamy, Raven, Octavia and Jasper deal with a vengeful Murphy. On the Ark, Jaha makes a plan to try and get to Earth.',

43, TO_DATE('4-Jul-2018', 'DD MONTH YYYY'), 8.4, 9);

INSERT INTO episodes

VALUES (16, 10, 'Who's da New King of Hell?', 'With murderous demons on the loose in Los Angeles, it's up to Lucifer to rein in the chaos and protect the ones he most cares about.', 55, TO_DATE('8-May-2019', 'DD MONTH YYYY'), 9.7, 12);

INSERT INTO episodes

VALUES (17, 7, 'Devil Is as Devil Does', 'Eve takes a more active role in her main man's professional life. Meanwhile, Lucifer gets back to basics and Amenadiel fights for his family.', 47, TO_DATE('8-May-2019', 'DD MONTH YYYY'), 9.3, 12);

INSERT INTO episodes

VALUES (18, 13, 'Take Me Back to Hell', 'When Lucifer is framed for murder, he and Chloe must work together to clear his name and prove the identity of the true killer.', 43, TO_DATE('25-Apr-2016', 'DD MONTH YYYY'), 9.2, 11);

EPISODE_ID	EPISODE_NUMBER	TITLE	DESCRIPTION
1	1	22Swan Song	With the Apocalypse looming
2	2	8Changing Channels	Sam and Dean catch up with
3	3	19Inherit the Earth	Everything is on the line
4	4	10New York, I Love You XOXO	In a fashionable farewell
5	5	7Victor/Victrola	Serena and Dan finally ad
6	6	22From a Cradle to a Grave	As the baby's due date dra
7	7	22The Bloody Crown	After months of thwarting
8	8	14A Streetcar Named Desire	The unexpected arrival of
9	9	9The Climb	The League of Assassins q
10	10	23Unthinkable	As Oliver's face off with
11	11	18Deathstroke	After taking a ride home
12	12	13Damocles: Part Two	Clarke and her friends mus
13	13	12Damocles: Part One	In part one of the fifth s
14	14	12We Are Grounders: Part 2	As the remaining members o
15	15	11We Are Grounders: Part 1	Clarke and Finn come face
16	16	10Who's da New King of Hell?	With murderous demons on
17	17	7Devil Is as Devil Does	Eve takes a more active ro
18	18	13Take Me Back to Hell	When Lucifer is framed fo

6) Tabela actors

INSERT INTO actors

VALUES (1, 'Jared', 'Padalecki', '19-JUL-1982');

INSERT INTO actors

VALUES (2, 'Jensen', 'Ackles', '01-MAR-1978');

INSERT INTO actors

VALUES (3, 'Misha', 'Collins', '20-AUG-1974');


```
INSERT INTO actors  
VALUES (4, 'Mark', 'Sheppard', '30-MAY-1964');
```

```
INSERT INTO actors  
VALUES (5, 'Alexander', 'Calvert', '15-JUL-1990');
```

```
INSERT INTO actors  
VALUES (6, 'Rob', 'Benedict', '21-SEP-1970');
```

```
INSERT INTO actors  
VALUES (7, 'Blake', 'Lively', '25-AUG-1987');
```

```
INSERT INTO actors  
VALUES (8, 'Leighton', 'Master', '09-APR-1986');
```

```
INSERT INTO actors  
VALUES (9, 'Penn', 'Badgley', '01-NOV-1986');
```

```
INSERT INTO actors  
VALUES (10, 'Ed', 'Westwick', '27-JUN-1987');
```

```
INSERT INTO actors  
VALUES (11, 'Chace', 'Crawford', '18-JUL-1985');
```

```
INSERT INTO actors  
VALUES (12, 'Joseph', 'Morgan', '16-MAY-1981');
```

```
INSERT INTO actors  
VALUES (13, 'Daniel', 'Gilles', '14-MAR-1976');
```

```
INSERT INTO actors  
VALUES (14, 'Claire', 'Holt', '11-JUN-1988');
```

```
INSERT INTO actors  
VALUES (15, 'Riley', 'Voelkel', '26-APR-1990');
```

```
INSERT INTO actors  
VALUES (16, 'Nathaniel', 'Buzolic', '04-AUG-1983');
```

```
INSERT INTO actors  
VALUES (17, 'Danielle Rose', 'Russell', '31-OCT-1999');
```

```
INSERT INTO actors  
VALUES (18, 'Phoebe', 'Tonkin', '12-JUL-1989');
```

```
INSERT INTO actors  
VALUES (19, 'Charles Michael', 'Davis', '01-DEC-1984');
```

```
INSERT INTO actors  
VALUES (20, 'Stephen', 'Amell', '08-MAY-1981');
```

```
INSERT INTO actors  
VALUES (21, 'Willa', 'Holland', '18-JUN-1991');
```

```
INSERT INTO actors  
VALUES (22, 'Emily Bett', 'Rickards', '24-JUL-1991');
```

```
INSERT INTO actors  
VALUES (23, 'David', 'Ramsey', '17-NOV-1971');
```

```
INSERT INTO actors  
VALUES (24, 'Katie', 'Cassidy', '25-NOV-1986');
```

```
INSERT INTO actors  
VALUES (25, 'Manu', 'Bennett', '10-OCT-1969');
```

```
INSERT INTO actors  
VALUES (26, 'Eliza', 'Taylor', '24-OCT-1989');
```

```
INSERT INTO actors  
VALUES (27, 'Marie', 'Avgeropoulos', '17-JUN-1986');
```

```
INSERT INTO actors  
VALUES (28, 'Bob', 'Morley', '20-DEC-1984');
```

```
INSERT INTO actors  
VALUES (29, 'Lindsey', 'Morgan', '27-FEB-1990');
```

```
INSERT INTO actors  
VALUES (30, 'Richard', 'Harmon', '18-AUG-1991');
```

```
INSERT INTO actors  
VALUES (31, 'Cristopher', 'Larkin', '02-OCT-1987');
```

```
INSERT INTO actors
VALUES (32, 'Tom', 'Ellis', '17-NOV-1978');
```

```
INSERT INTO actors
VALUES (33, 'Lauren', 'German', '29-NOV-1978');
```

```
INSERT INTO actors
VALUES (34, 'David Bryan', 'Woodside', '25-JUL-1969');
```

```
INSERT INTO actors
VALUES (35, 'Lesley-Ann', 'Brandt', '02-DEC-1981');
```

```
INSERT INTO actors
VALUES (36, 'Rachel', 'Harris', '12-JAN-1968');
```

	⚡ ACTOR_ID	⚡ FIRST_NAME	⚡ LAST_NAME	⚡ BIRTH_DATE
1	1	Jared	Padalecki	19-JUL-82
2	2	Jensen	Ackles	01-MAR-78
3	3	Misha	Collins	20-AUG-74
4	4	Mark	Sheppard	30-MAY-64
5	5	Alexander	Calvert	15-JUL-90
6	6	Rob	Benedict	21-SEP-70
7	7	Blake	Lively	25-AUG-87
8	8	Leighton	Master	09-APR-86
9	9	Penn	Badgley	01-NOV-86
10	10	Ed	Westwick	27-JUN-87
11	11	Chace	Crawford	18-JUL-85
12	12	Joseph	Morgan	16-MAY-81
13	13	Daniel	Gilles	14-MAR-76
14	14	Claire	Holt	11-JUN-88
15	15	Riley	Voelkel	26-APR-90
16	16	Nathaniel	Buzolic	04-AUG-83
17	17	Danielle Rose	Russell	31-OCT-99
18	18	Phoebe	Tonkin	12-JUL-89
19	19	Charles Michael	Davis	01-DEC-84
20	20	Stephen	Amell	08-MAY-81

7) Tabela characters

```
INSERT INTO characters
VALUES (1, 'Sam', 'Winchester', 1);
```

```
INSERT INTO characters
VALUES (2, 'Dean', 'Winchester', 1);
```

```
INSERT INTO characters
```

```
VALUES (3, 'Castiel', null, 1);
```

```
INSERT INTO characters
```

```
VALUES (4, 'Crowley', null, 1);
```

```
INSERT INTO characters
```

```
VALUES (5, 'Jack', null, 1);
```

```
INSERT INTO characters
```

```
VALUES (6, 'God', null, 1);
```

```
INSERT INTO characters
```

```
VALUES (7, 'Serena', 'van der Woodsen' , 2);
```

```
INSERT INTO characters
```

```
VALUES (8, 'Blair', 'Waldorf', 2);
```

```
INSERT INTO characters
```

```
VALUES (9, 'Dan', 'Humphrey', 2);
```

```
INSERT INTO characters
```

```
VALUES (10, 'Chuck', 'Bass', 2);
```

```
INSERT INTO characters
```

```
VALUES (11, 'Nate', 'Archibald', 2);
```

```
INSERT INTO characters
```

```
VALUES (12, 'Klaus', 'Mikaelson', 3);
```

```
INSERT INTO characters
```

```
VALUES (13, 'Elijah', 'Mikaelson', 3);
```

```
INSERT INTO characters
```

```
VALUES (14, 'Rebekah', 'Mikaelson', 3);
```

```
INSERT INTO characters
```

```
VALUES (15, 'Freya', 'Mikaelson', 3);
```

```
INSERT INTO characters
```

```
VALUES (16, 'Kol', 'Mikaelson', 3);
```

```
INSERT INTO characters
```

```
VALUES (17, 'Hope', 'Mikaelson', 3);
```

```
INSERT INTO characters
```

```
VALUES (18, 'Hayley', 'Marshall', 3);
```

```
INSERT INTO characters
```

```
VALUES (19, 'Marcel', 'Gerard', 3);
```

```
INSERT INTO characters
```

```
VALUES (20, 'Oliver', 'Queen', 4);
```

```
INSERT INTO characters
```

```
VALUES (21, 'Thea', 'Queen', 4);
```

```
INSERT INTO characters
```

```
VALUES (22, 'Felicity', 'Smoak', 4);
```

```
INSERT INTO characters
```

```
VALUES (23, 'John', 'Diggle', 4);
```

```
INSERT INTO characters
```

```
VALUES (24, 'Laurel', 'Lance', 4);
```

```
INSERT INTO characters
```

```
VALUES (25, 'Slade', 'Wilson', 4);
```

```
INSERT INTO characters
```

```
VALUES (26, 'Clark', 'Griffin', 5);
```

```
INSERT INTO characters
```

```
VALUES (27, 'Octavia', 'Blake', 5);
```

```
INSERT INTO characters
```

```
VALUES (28, 'Bellamy', 'Blake', 5);
```

```
INSERT INTO characters
```

```
VALUES (29, 'Raven', 'Reyes', 5);
```

```
INSERT INTO characters
```

```
VALUES (30, 'John', 'Murphy', 5);
```

```
INSERT INTO characters
```

```
VALUES (31, 'Monty', 'Green', 5);
```

```
INSERT INTO characters
```

```
VALUES (32, 'Lucifer', 'Morningstar', 6);
```

```
INSERT INTO characters
```

```
VALUES (33, 'Chloe', 'Decker', 6);
```

```
INSERT INTO characters
```

```
VALUES (34, 'Amenadiel', null, 6);
```

```
INSERT INTO characters
```

```
VALUES (35, 'Mazikeen', null, 6);
```

```
INSERT INTO characters
```

```
VALUES (36, 'Linda', 'Martin', 6);
```

	CHARACTER_ID	FIRST_NAME	LAST_NAME	SERIES_ID
1	1	Sam	Winchester	1
2	2	Dean	Winchester	1
3	3	Castiel	(null)	1
4	4	Crowley	(null)	1
5	5	Jack	(null)	1
6	6	God	(null)	1
7	7	Serena	van der Woodsen	2
8	8	Blair	Waldorf	2
9	9	Dan	Humphrey	2
10	10	Chuck	Bass	2
11	11	Nate	Archibald	2
12	12	Klaus	Mikaelson	3
13	13	Elijah	Mikaelson	3
14	14	Rebekah	Mikaelson	3
15	15	Freya	Mikaelson	3
16	16	Kol	Mikaelson	3
17	17	Hope	Mikaelson	3
18	18	Hayley	Marshall	3
19	19	Marcel	Gerard	3
20	20	Oliver	Queen	4

8) Tabela playing

```
INSERT INTO playing
```

```
VALUES (1, 1, 1, TO_DATE('2005', 'YYYY'), TO_DATE('2020', 'YYYY'), 327);
```

```
INSERT INTO playing
```

```
VALUES (1, 2, 2, TO_DATE('2005', 'YYYY'), TO_DATE('2020', 'YYYY'), 327);
```

```
INSERT INTO playing  
VALUES (1, 3, 3, TO_DATE('2008', 'YYYY'), TO_DATE('2020', 'YYYY'), 148);
```

```
INSERT INTO playing  
VALUES (1, 4, 4, TO_DATE('2009', 'YYYY'), TO_DATE('2017', 'YYYY'), 72);
```

```
INSERT INTO playing  
VALUES (1, 5, 5, TO_DATE('2017', 'YYYY'), TO_DATE('2020', 'YYYY'), 39);
```

```
INSERT INTO playing  
VALUES (1, 6, 6, TO_DATE('2009', 'YYYY'), TO_DATE('2020', 'YYYY'), 9);
```

```
INSERT INTO playing  
VALUES (2, 7, 7, TO_DATE('2007', 'YYYY'), TO_DATE('2012', 'YYYY'), 121);
```

```
INSERT INTO playing  
VALUES (2, 8, 8, TO_DATE('2007', 'YYYY'), TO_DATE('2012', 'YYYY'), 121);
```

```
INSERT INTO playing  
VALUES (2, 9, 9, TO_DATE('2007', 'YYYY'), TO_DATE('2012', 'YYYY'), 121);
```

```
INSERT INTO playing  
VALUES (2, 10, 10, TO_DATE('2007', 'YYYY'), TO_DATE('2012', 'YYYY'), 121);
```

```
INSERT INTO playing  
VALUES (2, 11, 11, TO_DATE('2007', 'YYYY'), TO_DATE('2012', 'YYYY'), 121);
```

```
INSERT INTO playing  
VALUES (3, 12, 12, TO_DATE('2013', 'YYYY'), TO_DATE('2018', 'YYYY'), 92);
```

```
INSERT INTO playing  
VALUES (3, 13, 13, TO_DATE('2013', 'YYYY'), TO_DATE('2018', 'YYYY'), 92);
```

```
INSERT INTO playing  
VALUES (3, 14, 14, TO_DATE('2013', 'YYYY'), TO_DATE('2018', 'YYYY'), 40);
```

```
INSERT INTO playing  
VALUES (3, 15, 15, TO_DATE('2014', 'YYYY'), TO_DATE('2018', 'YYYY'), 60);
```

```
INSERT INTO playing  
VALUES (3, 16, 16, TO_DATE('2013', 'YYYY'), TO_DATE('2018', 'YYYY'), 24);
```

```
INSERT INTO playing  
VALUES (3, 17, 17, TO_DATE('2018', 'YYYY'), TO_DATE('2018', 'YYYY'), 13);
```

```
INSERT INTO playing  
VALUES (3, 18, 18, TO_DATE('2013', 'YYYY'), TO_DATE('2018', 'YYYY'), 86);
```

```
INSERT INTO playing  
VALUES (3, 19, 19, TO_DATE('2013', 'YYYY'), TO_DATE('2018', 'YYYY'), 92);
```

```
INSERT INTO playing  
VALUES (4, 20, 20, TO_DATE('2012', 'YYYY'), TO_DATE('2020', 'YYYY'), 170);
```

```
INSERT INTO playing  
VALUES (4, 21, 21, TO_DATE('2012', 'YYYY'), TO_DATE('2020', 'YYYY'), 134);
```

```
INSERT INTO playing  
VALUES (4, 22, 22, TO_DATE('2012', 'YYYY'), TO_DATE('2020', 'YYYY'), 157);
```

```
INSERT INTO playing  
VALUES (4, 23, 23, TO_DATE('2012', 'YYYY'), TO_DATE('2020', 'YYYY'), 170);
```

```
INSERT INTO playing  
VALUES (4, 24, 24, TO_DATE('2012', 'YYYY'), TO_DATE('2020', 'YYYY'), 153);
```

```
INSERT INTO playing  
VALUES (4, 25, 25, TO_DATE('2013', 'YYYY'), TO_DATE('2017', 'YYYY'), 40);
```

```
INSERT INTO playing  
VALUES (5, 26, 26, TO_DATE('2014', 'YYYY'), TO_DATE('2020', 'YYYY'), 100);
```

```
INSERT INTO playing  
VALUES (5, 27, 27, TO_DATE('2014', 'YYYY'), TO_DATE('2020', 'YYYY'), 100);
```

```
INSERT INTO playing  
VALUES (5, 28, 28, TO_DATE('2014', 'YYYY'), TO_DATE('2020', 'YYYY'), 97);
```

```
INSERT INTO playing  
VALUES (5, 29, 29, TO_DATE('2014', 'YYYY'), TO_DATE('2020', 'YYYY'), 98);
```

```
INSERT INTO playing  
VALUES (5, 30, 30, TO_DATE('2014', 'YYYY'), TO_DATE('2020', 'YYYY'), 90);
```



```
INSERT INTO playing
VALUES (5, 31, 31, TO_DATE('2014', 'YYYY'), TO_DATE('2019', 'YYYY'), 73);
```

```
INSERT INTO playing
VALUES (6, 32, 32, TO_DATE('2016', 'YYYY'), null, 78);
```

```
INSERT INTO playing
VALUES (6, 33, 33, TO_DATE('2016', 'YYYY'), null, 78);
```

```
INSERT INTO playing
VALUES (6, 34, 34, TO_DATE('2016', 'YYYY'), null, 77);
```

```
INSERT INTO playing
VALUES (6, 35, 35, TO_DATE('2016', 'YYYY'), null, 77);
```

```
INSERT INTO playing
VALUES (6, 36, 36, TO_DATE('2016', 'YYYY'), null, 77);
```

	SERIES_ID	ACTOR_ID	CHARACTER_ID	STARTING_DATE	ENDING_DATE	EPISODES_NUMBER
1	1	1	1	01-DEC-05	01-DEC-20	327
2	1	2	2	01-DEC-05	01-DEC-20	327
3	1	3	3	01-DEC-08	01-DEC-20	148
4	1	4	4	01-DEC-09	01-DEC-17	72
5	1	5	5	01-DEC-17	01-DEC-20	39
6	1	6	6	01-DEC-09	01-DEC-20	9
7	2	7	7	01-DEC-07	01-DEC-12	121
8	2	8	8	01-DEC-07	01-DEC-12	121
9	2	9	9	01-DEC-07	01-DEC-12	121
10	2	10	10	01-DEC-07	01-DEC-12	121
11	2	11	11	01-DEC-07	01-DEC-12	121
12	3	12	12	01-DEC-13	01-DEC-18	92
13	3	13	13	01-DEC-13	01-DEC-18	92
14	3	14	14	01-DEC-13	01-DEC-18	40
15	3	15	15	01-DEC-14	01-DEC-18	60
16	3	16	16	01-DEC-13	01-DEC-18	24
17	3	17	17	01-DEC-18	01-DEC-18	13
18	3	18	18	01-DEC-13	01-DEC-18	86
19	3	19	19	01-DEC-13	01-DEC-18	92
20	4	20	20	01-DEC-12	01-DEC-20	170

9) Tabela appearing_in

```
INSERT INTO appearing_in
VALUES (1, 1);
```

```
INSERT INTO appearing_in  
VALUES (1, 2);
```

```
INSERT INTO appearing_in  
VALUES (1, 3);
```

```
INSERT INTO appearing_in  
VALUES (2, 1);
```

```
INSERT INTO appearing_in  
VALUES (2, 2);
```

```
INSERT INTO appearing_in  
VALUES (2, 3);
```

```
INSERT INTO appearing_in  
VALUES (3, 1);
```

```
INSERT INTO appearing_in  
VALUES (3, 2);
```

```
INSERT INTO appearing_in  
VALUES (3, 3);
```

```
INSERT INTO appearing_in  
VALUES (3, 5);
```

```
INSERT INTO appearing_in  
VALUES (3, 6);
```

```
INSERT INTO appearing_in  
VALUES (4, 7);
```

```
INSERT INTO appearing_in  
VALUES (4, 8);
```

```
INSERT INTO appearing_in  
VALUES (4, 9);
```

```
INSERT INTO appearing_in  
VALUES (4, 10);
```

```
INSERT INTO appearing_in  
VALUES (4, 11);
```

```
INSERT INTO appearing_in  
VALUES (5, 7);
```

```
INSERT INTO appearing_in  
VALUES (5, 8);
```

```
INSERT INTO appearing_in  
VALUES (5, 9);
```

```
INSERT INTO appearing_in  
VALUES (5, 10);
```

```
INSERT INTO appearing_in  
VALUES (5, 11);
```

```
INSERT INTO appearing_in  
VALUES (6, 12);
```

```
INSERT INTO appearing_in  
VALUES (6, 13);
```

```
INSERT INTO appearing_in  
VALUES (6, 14);
```

```
INSERT INTO appearing_in  
VALUES (6, 18);
```

```
INSERT INTO appearing_in  
VALUES (6, 19);
```

```
INSERT INTO appearing_in  
VALUES (7, 12);
```

```
INSERT INTO appearing_in  
VALUES (7, 13);
```

```
INSERT INTO appearing_in  
VALUES (7, 14);
```

```
INSERT INTO appearing_in  
VALUES (7, 15);
```

```
INSERT INTO appearing_in  
VALUES (7, 16);
```

```
INSERT INTO appearing_in  
VALUES (7, 18);
```

```
INSERT INTO appearing_in  
VALUES (7, 19);
```

```
INSERT INTO appearing_in  
VALUES (8, 12);
```

```
INSERT INTO appearing_in  
VALUES (8, 13);
```

```
INSERT INTO appearing_in  
VALUES (8, 15);
```

```
INSERT INTO appearing_in  
VALUES (8, 16);
```

```
INSERT INTO appearing_in  
VALUES (8, 18);
```

```
INSERT INTO appearing_in  
VALUES (8, 19);
```

```
INSERT INTO appearing_in  
VALUES (9, 20);
```

```
INSERT INTO appearing_in  
VALUES (9, 21);
```

```
INSERT INTO appearing_in  
VALUES (9, 22);
```

```
INSERT INTO appearing_in  
VALUES (9, 23);
```

```
INSERT INTO appearing_in  
VALUES (9, 24);
```

```
INSERT INTO appearing_in  
VALUES (10, 20);
```

```
INSERT INTO appearing_in  
VALUES (10, 21);
```

```
INSERT INTO appearing_in  
VALUES (10, 22);
```

```
INSERT INTO appearing_in  
VALUES (10, 23);
```

```
INSERT INTO appearing_in  
VALUES (10, 24);
```

```
INSERT INTO appearing_in  
VALUES (10, 25);
```

```
INSERT INTO appearing_in  
VALUES (11, 20);
```

```
INSERT INTO appearing_in  
VALUES (11, 21);
```

```
INSERT INTO appearing_in  
VALUES (11, 22);
```

```
INSERT INTO appearing_in  
VALUES (11, 23);
```

```
INSERT INTO appearing_in  
VALUES (11, 24);
```

```
INSERT INTO appearing_in  
VALUES (11, 25);
```

```
INSERT INTO appearing_in  
VALUES (12, 26);
```

```
INSERT INTO appearing_in  
VALUES (12, 27);
```

```
INSERT INTO appearing_in  
VALUES (12, 28);
```

```
INSERT INTO appearing_in  
VALUES (12, 29);
```

```
INSERT INTO appearing_in  
VALUES (12, 30);
```

```
INSERT INTO appearing_in  
VALUES (12, 31);
```

```
INSERT INTO appearing_in  
VALUES (13, 26);
```

```
INSERT INTO appearing_in  
VALUES (13, 27);
```

```
INSERT INTO appearing_in  
VALUES (13, 28);
```

```
INSERT INTO appearing_in  
VALUES (13, 29);
```

```
INSERT INTO appearing_in  
VALUES (13, 30);
```

```
INSERT INTO appearing_in  
VALUES (13, 31);
```

```
INSERT INTO appearing_in  
VALUES (14, 26);
```

```
INSERT INTO appearing_in  
VALUES (14, 27);
```

```
INSERT INTO appearing_in  
VALUES (14, 28);
```

```
INSERT INTO appearing_in  
VALUES (14, 29);
```

```
INSERT INTO appearing_in  
VALUES (14, 30);
```

```
INSERT INTO appearing_in  
VALUES (14, 31);
```

```
INSERT INTO appearing_in  
VALUES (15, 26);
```

```
INSERT INTO appearing_in  
VALUES (15, 27);
```

```
INSERT INTO appearing_in  
VALUES (15, 28);
```

```
INSERT INTO appearing_in  
VALUES (15, 29);
```

```
INSERT INTO appearing_in  
VALUES (15, 30);
```

```
INSERT INTO appearing_in  
VALUES (15, 31);
```

```
INSERT INTO appearing_in  
VALUES (16, 32);
```

```
INSERT INTO appearing_in  
VALUES (16, 33);
```

```
INSERT INTO appearing_in  
VALUES (16, 34);
```

```
INSERT INTO appearing_in  
VALUES (16, 35);
```

```
INSERT INTO appearing_in  
VALUES (16, 36);
```

```
INSERT INTO appearing_in
VALUES (17, 32);
```

```
INSERT INTO appearing_in
VALUES (17, 33);
```

```
INSERT INTO appearing_in
VALUES (17, 34);
```

```
INSERT INTO appearing_in
VALUES (17, 35);
```

```
INSERT INTO appearing_in
VALUES (17, 36);
```

```
INSERT INTO appearing_in
VALUES (18, 32);
```

```
INSERT INTO appearing_in
VALUES (18, 33);
```

```
INSERT INTO appearing_in
VALUES (18, 34);
```

```
INSERT INTO appearing_in
VALUES (18, 35);
```

```
INSERT INTO appearing_in
VALUES (18, 36);
```

	EPISODE_ID	CHARACTER_ID
1	1	1
2	1	2
3	1	3
4	2	1
5	2	2
6	2	3
7	3	1
8	3	2
9	3	3
10	3	5
11	2	6

6. Definirea unui subprogram stocat care să utilizeze un tip de colecție studiat (tablou imbricat)

Sa se modifice lista de categorii a unui serial specificat:

- i) adăugarea unei categorii (se da valoarea noii categorii plus cuvântul 'INSERTING')
- ii) ștergerea unei categorii (se da valoarea categoriei plus cuvântul 'DELETING')
- iii) modificarea unei categorii (se da valoarea vechii categorii, noii categorii plus cuvântul 'UPDATING')

```
CREATE OR REPLACE PROCEDURE modificare_categorii
(serial series.title%TYPE,
categ1 VARCHAR2,
optiune VARCHAR2,
categ2 VARCHAR2 := NULL)
AS
  categorii genres;
  i INTEGER;
BEGIN
  -- obtinere lista categorii pentru serialul dat
  SELECT genre INTO categorii
  FROM series
  WHERE title = INITCAP(serial);

  IF UPPER(optiune) = 'INSERTING' THEN
    IF categ1 IS NULL THEN
      RAISE_APPLICATION_ERROR(-20001, 'Categoria nou introdusa nu poate sa fie
NULL');
    ELSE
      -- adaugarea unei categorii noi
      categorii.extend();
      categorii(categorii.last) := INITCAP(categ1);
    END IF;
  ELSIF UPPER(optiune) = 'DELETING' THEN
    IF categ1 IS NULL THEN
      RAISE_APPLICATION_ERROR(-20001, 'Categoria de sters nu poate sa fie NULL');
    ELSE
      -- determinarea pozitiei categoriei ce trebuie stersa
      i := categorii.FIRST;
      WHILE i <= categorii.LAST AND categorii(i) <> INITCAP(categ1) LOOP
        i := categorii.NEXT(i);
```

```
END LOOP;

IF i IS NOT NULL THEN
    -- stergerea categoriei
    categorii.DELETE(i);
ELSE
    RAISE_APPLICATION_ERROR(-20002, 'Nu exista categoria introdusa');
END IF;
END IF;
ELSIF UPPER(optiune) = 'UPDATING' THEN
    IF categ1 IS NULL THEN
        RAISE_APPLICATION_ERROR(-20001, 'Categoria de actualizat nu poate sa fie
NULL');
    ELSE
        IF categ2 IS NULL THEN
            RAISE_APPLICATION_ERROR(-20001, 'Categoria nou introdusa nu poate sa fie
NULL');
        ELSE
            -- determinarea pozitiei categoriei ce trebuie modificata
            i := categorii.FIRST;
            WHILE i <= categorii.LAST AND categorii(i) <> INITCAP(categ1) LOOP
                i := categorii.NEXT(i);
            END LOOP;

            IF i IS NOT NULL THEN
                -- odificarea categoriei
                categorii(i) := INITCAP(categ2);
            ELSE
                RAISE_APPLICATION_ERROR(-20002, 'Nu exista categoria introdusa');
            END IF;
        END IF;
    END IF;
ELSE
    RAISE_APPLICATION_ERROR(-20003, 'Optiunea introdusa este gresita');
END IF;

-- actualizare lista categorii
UPDATE series
SET genre = categorii
WHERE title = serial;

DBMS_OUTPUT.PUT_LINE('Lista de categorii a fost actualizata cu succes');
```

EXCEPTION

```

WHEN no_data_found THEN
    RAISE_APPLICATION_ERROR(-20004, 'Nu exista serial cu numele dat');
WHEN too_many_rows THEN
    RAISE_APPLICATION_ERROR(-20005, 'Exista mai multe seriale cu acest nume');
END;
/

```

The screenshot shows the SQL Developer interface. The 'Query Builder' tab is active, displaying a SQL script with a merge statement and an EXECUTE statement for updating categories. The 'Script Output' tab shows the message: 'Lista de categorii a fost actualizata cu succes'.

The screenshot shows the SQL Developer interface. The 'Query Builder' tab is active, displaying a SQL script with a merge statement and an EXECUTE statement for updating categories. The 'Script Output' tab shows an error message: 'Error starting at line : 92 in command - BEGIN modificare_categorii('The Magicians', null, 'inserting'); END; Error report - ORA-20001: Categoria nou introdusa nu poate sa fie NULL ORA-06512: at "C##CLAUDIA.MODIFICARE_CATEGORII", line 17 ORA-06512: at line 1'.

The screenshot shows the SQL Developer interface. The 'Query Builder' tab is active, displaying a SQL script with a merge statement and an EXECUTE statement for deleting a category. The 'Script Output' tab shows the message: 'Lista de categorii a fost actualizata cu succes'.

The screenshot shows the SQL Developer interface. The 'Query Builder' tab is active, displaying a SQL script with a merge statement and an EXECUTE statement for deleting a category. The 'Script Output' tab shows an error message: 'Error starting at line : 106 in command - BEGIN modificare_categorii('Supernatural', 'Action', 'deleting'); END; Error report - ORA-20002: Nu exista categoria introdusa ORA-06512: at "C##CLAUDIA.MODIFICARE_CATEGORII", line 37 ORA-06512: at line 1'.

```
-- eroare: categoria de sters nu poate fi null
EXECUTE modificare_categorii('Supernatural', null, 'deleting');
```

Script Output x

Task completed in 0.083 seconds

Error starting at line : 109 in command -
BEGIN modificare_categorii('Supernatural', null, 'deleting'); END;
Error report -
ORA-20001: Categoria de sters nu poate sa fie NULL
ORA-06512: at "C##CLAUDIA.MODIFICARE_CATEGORII", line 25
ORA-06512: at line 1

```
-- eroare: categoria cu care se inlocuieste nu poate fi null
EXECUTE modificare_categorii('Arrow', 'Action', 'updating');
```

Script Output x

Task completed in 0.095 seconds

Error starting at line : 120 in command -
BEGIN modificare_categorii('Arrow', 'Action', 'updating'); END;
Error report -
ORA-20001: Categoria nou introdusa nu poate sa fie NULL
ORA-06512: at "C##CLAUDIA.MODIFICARE_CATEGORII", line 45
ORA-06512: at line 1

Worksheet Query Builder

```
-- merge
EXECUTE modificare_categorii('Arrow', 'Action', 'updating', 'Drama');
```

Lista de categorii a fost actualizata cu succes

```
-- eroare: categoria de inlocuit nu poate fi null
EXECUTE modificare_categorii('Arrow', null, 'updating', 'Drama');
```

Script Output x

Task completed in 0.133 seconds

Error starting at line : 126 in command -
BEGIN modificare_categorii('Arrow', null, 'updating', 'Drama'); END;
Error report -
ORA-20001: Categoria de actualizat nu poate sa fie NULL
ORA-06512: at "C##CLAUDIA.MODIFICARE_CATEGORII", line 42
ORA-06512: at line 1

```
-- eroare: nu exista categoria de modificat
EXECUTE modificare_categorii('Arrow', 'Thriller', 'updating', 'Drama');
```

Script Output x

Task completed in 0.145 seconds

```
Error starting at line : 129 in command -
BEGIN modificare_categorii('Arrow', 'Thriller', 'updating', 'Drama'); END;
Error report -
ORA-20002: Nu exista categoria introdusa
ORA-06512: at "C##CLAUDIA.MODIFICARE_CATEGORII", line 57
ORA-06512: at line 1
```

```
-- eroare: comanda invalida
EXECUTE modificare_categorii('Arrow', 'Drama', 'insert');
```

Script Output x

Task completed in 0.276 seconds

```
Error starting at line : 136 in command -
BEGIN modificare_categorii('Arrow', 'Drama', 'insert'); END;
Error report -
ORA-20003: Optiunea introdusa este gresita
ORA-06512: at "C##CLAUDIA.MODIFICARE_CATEGORII", line 62
ORA-06512: at line 1
```

7. Definirea unui subprogram stocat care să utilizeze un tip de cursor studiat (expresii cursor)

Sa se afișeze toate sezoanele si episoadele unui serial specificat (sezon: număr, data început, data sfârșit; episod: număr, nume, descriere, rating).

```
CREATE OR REPLACE PROCEDURE afisare_episoade
(serial series.title%TYPE)
AS
TYPE ref_cursor IS REF CURSOR;
CURSOR sezoane (id_serial NUMBER) IS
SELECT season_number, starting_date, ending_date,
CURSOR (SELECT episode_number, title, description, rating
FROM episodes e
```

```

        WHERE e.season_id = s.season_id)
    FROM seasons s
    WHERE series_id = id_serial;
episoade ref_cursor;
id_serial series.series_id%TYPE;
numar_sez seasons.season_number%TYPE;
inceput_sez seasons.starting_date%TYPE;
sfarsit_sez seasons.ending_date%TYPE;
TYPE ep IS RECORD (numar episodes.episode_number%TYPE,
                    titlu episodes.title%TYPE,
                    descriere episodes.description%TYPE,
                    rating episodes.rating%TYPE);
episod ep;
exista_sezoane BOOLEAN := FALSE;
exista_episoade BOOLEAN;
BEGIN
    -- determina id serial
    SELECT series_id INTO id_serial
    FROM series
    WHERE title = serial;

    OPEN sezoane(id_serial);
    LOOP
        FETCH sezoane INTO numar_sez, inceput_sez, sfarsit_sez, episoade;
        EXIT WHEN sezoane%NOTFOUND;
        exista_sezoane := TRUE;

        -- afisare sezon
        DBMS_OUTPUT.PUT_LINE('-----');
        DBMS_OUTPUT.PUT('SEZONUL ' || numar_sez || ': ' || inceput_sez || ' - ');

        IF sfarsit_sez IS NULL THEN
            -- sezonul se afla in derulare
            DBMS_OUTPUT.PUT_LINE('prezent');
        ELSE
            DBMS_OUTPUT.PUT_LINE(sfarsit_sez);
        END IF;

        -- afisare episoade
        exista_episoade := FALSE;

    LOOP

```

```

    FETCH episoade INTO episod;
    EXIT WHEN episoade%NOTFOUND;

    exista_episoade := TRUE;

    DBMS_OUTPUT.PUT_LINE(episod.numar || ' ' || episod.titlu || ' - ' || episod.rating);
    DBMS_OUTPUT.PUT_LINE('Synopsis: ' || episod.descriere);
    DBMS_OUTPUT.PUT_LINE("");
END LOOP;

IF NOT exista_episoade THEN
    DBMS_OUTPUT.PUT_LINE('Nu exista episoade pentru acest sezon');
END IF;

    DBMS_OUTPUT.PUT_LINE('-----');
    DBMS_OUTPUT.PUT_LINE("");
END LOOP;
CLOSE sezoane;

IF NOT exista_sezoane THEN
    DBMS_OUTPUT.PUT_LINE('Nu exista sezoane pentru acest serial');
END IF;
EXCEPTION
    WHEN no_data_found THEN
        RAISE_APPLICATION_ERROR(-20004, 'Nu exista serial cu numele dat');
    WHEN too_many_rows THEN
        RAISE_APPLICATION_ERROR(-20005, 'Exista mai multe seriale cu acest nume');
END;
/

```

The screenshot shows an SQL Worksheet interface with a 'Script Output' window at the bottom. The main window displays the following SQL code:

```
-- merge
EXECUTE afisare_episoade('The 100');
```

The 'Script Output' window shows the following output:

```
PL/SQL procedure successfully completed.

-----
SEZONUL 1: 19-MAR-14 - 11-JUN-14
12. We Are Grounders: Part 2 - 8.9
Synopsis: As the remaining members of the 100 face off
-----
11. We Are Grounders: Part 1 - 8.4
Synopsis: Clarke and Finn come face to face with a new
-----
SEZONUL 5: 24-APR-18 - 07-AUG-18
13. Damocles: Part Two - 9.6
Synopsis: Clarke and her friends must risk everything
-----
12. Damocles: Part One - 9
Synopsis: In part one of the fifth season finale, Octa
-----
```

The first screenshot shows a SQL query in a 'Query Builder' window. The query is:


```
-- merge
EXECUTE afisare_episoade('The Magicians');
```

 The output pane on the right displays the message: "Nu exista sezoane pentru acest serial".

The second screenshot shows the same query editor with an additional SQL statement:


```
INSERT INTO seasons
VALUES (13, 1, '16-DEC-15', '11-APR-16', 7);
```

 followed by the merge command. The output pane shows:


```
SEZONUL 1: 16-DEC-15 - 11-APR-16
Nu exista episoade pentru acest sezon
```

The third screenshot shows the query editor with a new query:


```
-- eroare: nu exista serialul
EXECUTE afisare_episoade('Riverdale');
```

 Below the editor is a 'Script Output' window showing the execution status: "Task completed in 0.173 seconds". Below that, an error message is displayed:


```
Error starting at line : 238 in command -
BEGIN afisare_episoade('Riverdale'); END;
Error report -
ORA-20004: Nu exista serial cu numele dat
ORA-06512: at "C##CLAUDIA.AFISARE_EPISOADE", line 75
ORA-06512: at line 1
```

8. Definirea unei funcții care să utilizeze trei tabele diferite

Sa se determine numărul de episoade in care joaca un anumit actor într-o anumita perioada de timp.

```
CREATE OR REPLACE FUNCTION nr_episoade
(prenume actors.first_name%TYPE := NULL,
 nume actors.last_name%TYPE := NULL,
 inceput DATE,
 sfarsit DATE)
RETURN NUMBER IS
  nr_ep NUMBER(3);
  id_actor actors.actor_id%TYPE;
BEGIN
  IF inceput > sfarsit THEN
    RAISE_APPLICATION_ERROR(-20006, 'Data de inceput trebuie sa fie mai mica decat data
    de sfarsit');
```



```
        RETURN -1;
    END IF;

    IF prenume IS NULL AND nume IS NULL THEN
        RAISE_APPLICATION_ERROR(-20007, 'Nu poate sa fie si numele si prenumele NULL');
        RETURN -1;
    END IF;

    -- determinarea id-ului actorului dat
    -- (acest pas se face separat ca sa se poata arunca exceptie in cazul in care
    -- nu exista actorul sau exista mai multi actori cu acest nume)
    IF nume IS NOT NULL AND prenume IS NOT NULL THEN
        -- numele si prenumele nu sunt NULL
        SELECT actor_id INTO id_actor
        FROM actors
        WHERE first_name = prenume
        AND last_name = nume;
    ELSIF nume IS NULL AND prenume IS NOT NULL THEN
        -- prenumele nu este NULL
        SELECT actor_id INTO id_actor
        FROM actors
        WHERE first_name = prenume;
    ELSE
        -- numele nu este NULL
        SELECT actor_id INTO id_actor
        FROM actors
        WHERE last_name = nume;
    END IF;

    SELECT COUNT(*) INTO nr_ep
    FROM playing p JOIN characters ch ON (p.character_id = ch.character_id)
        JOIN appearing_in ap ON (ch.character_id = ap.character_id)
        JOIN episodes e ON (ap.episode_id = e.episode_id)
    WHERE p.actor_id = id_actor
        AND (p.starting_date <= sfarsit AND p.ending_date >= inceput)
        AND e.airing_date BETWEEN inceput AND sfarsit;

    RETURN nr_ep;
EXCEPTION
    WHEN no_data_found THEN
        RAISE_APPLICATION_ERROR(-20004, 'Nu exista actor cu numele dat');
```

```

RETURN -1;
WHEN too_many_rows THEN
    RAISE_APPLICATION_ERROR(-20005, 'Exista mai multi actori cu acest nume');
RETURN -1;
END;
/

```

```
-- merge
SELECT nr_episoade('Jared', 'Padalecki', '13-MAY-10', '12-NOV-20') FROM dual;
```

Script Output x Query Result x

SQL | All Rows Fetched: 1 in 0.117 seconds

NR_EPISODE('JARED','PADALECKI','13-MAY-10','12-NOV-20')
1

```
-- merge
SELECT nr_episoade('Joseph', null, '13-MAY-14', '20-MAY-16') FROM dual;
```

Script Output x Query Result x

SQL | All Rows Fetched: 1 in 0.015 seconds

NR_EPISODE('JOSEPH',NULL,'13-MAY-14','20-MAY-16')
3

```
-- merge
SELECT nr_episoade(null, 'Master', '23-OCT-10', '06-DEC-16') FROM dual;
```

Script Output x Query Result x

SQL | All Rows Fetched: 1 in 0.02 seconds

NR_EPISODE(NULL,'MASTER','23-OCT-10','06-DEC-16')
1

```
-- eroare: exista mai multi actori ci acest nume
SELECT nr_episoade(null, 'Morgan', '18-MAY-14', '19-APR-16') FROM dual;
```

Script Output x Query Result x

SQL | Executing: SELECT nr_episoade(null, 'Morgan', '18-MAY-14', '19-APR-16') FROM dual in 0 seconds

ORA-20005: Exista mai multi actori cu acest nume
ORA-06512: at "C##CLAUDIA.NR_EPISODE", line 56
ORA-06512: at line 1

```
-- eroare: nu exista actor cu acest nume
SELECT nr_episoade('John', 'Doe', '13-MAY-14', '20-MAY-16') FROM dual;
```

Script Output x Query Result x

SQL | Executing:SELECT nr_episoade('John', 'Doe', '13-MAY-14', '20-MAY-16') FROM dual in 0 seconds

ORA-20004: Nu exista actor cu numele dat
ORA-06512: at "C##CLAUDIA.NR_EPISOADE", line 53
ORA-06512: at line 1

```
-- eroare: numele si prenumele nu pot fi ambele null
SELECT nr_episoade(null, null, '13-APR-14', '25-Jan-15') FROM dual;
```

Script Output x Query Result x

SQL | Executing:SELECT nr_episoade(null, null, '13-APR-14', '25-Jan-15') FROM dual in 0 seconds

ORA-20007: Nu poate sa fie si numele si prenumele NULL
ORA-06512: at "C##CLAUDIA.NR_EPISOADE", line 16
ORA-06512: at line 1

```
-- eroare: data de inceput este mai mare dect data de sfarsit
SELECT nr_episoade('Emily Bett', 'Rickards', '13-SEP-14', '20-AUG-14') FROM dual;
```

Script Output x Query Result x

SQL | Executing:SELECT nr_episoade('Emily Bett', 'Rickards', '13-SEP-14', '20-AUG-14') FROM dual in 0 seconds

ORA-20006: Data de inceput trebuie sa fie mai mica decat data de sfarsit
ORA-06512: at "C##CLAUDIA.NR_EPISOADE", line 11
ORA-06512: at line 1

9. Definirea unei proceduri care să utilizeze cinci tabele diferite

Sa se afișeze pentru fiecare serial numele serialului, producătorii si lista actorilor împreuna cu personajele pe care le interpretează.

```
CREATE OR REPLACE PROCEDURE afisare_seriale AS
  TYPE pers IS RECORD (prenume producers.first_name%TYPE,
                        nume producers.last_name%TYPE);
  TYPE prod IS TABLE OF pers;
  v_producatori prod;
  CURSOR actori (id_serial NUMBER) IS
    SELECT a.actor_id, a.first_name, a.last_name
    FROM actors a JOIN playing p ON(a.actor_id = p.actor_id)
    WHERE series_id = id_serial;
  personaj pers;
```

```

i INTEGER;
exista_actori BOOLEAN;
BEGIN
  FOR serial IN (SELECT series_id, title
                  FROM series)
    LOOP
      -- afisare serial
      DBMS_OUTPUT.PUT_LINE('-----');
      DBMS_OUTPUT.PUT_LINE('--- ' || UPPER(serial.title) || ' ---');

      -- afisare producatori
      DBMS_OUTPUT.PUT('--- Producatori: ');

      SELECT p.first_name, p.last_name BULK COLLECT INTO v_producatori
      FROM producers p JOIN produced_by ps ON (p.producer_id = ps.producer_id)
      WHERE series_id = serial.series_id;

      IF v_producatori.count() = 0 THEN
        -- nu exista producatori
        DBMS_OUTPUT.PUT('nu exista producatori');
      ELSE
        i := v_producatori.FIRST;
        WHILE i <= v_producatori.LAST LOOP
          DBMS_OUTPUT.PUT(v_producatori(i).prenume || ' ' || v_producatori(i).nume);
          IF i <> v_producatori.LAST THEN
            DBMS_OUTPUT.PUT(', ');
          END IF;

          i := v_producatori.NEXT(i);
        END LOOP;
      END IF;

      DBMS_OUTPUT.PUT(' ---');
      DBMS_OUTPUT.PUT_LINE('');
      DBMS_OUTPUT.PUT_LINE('-----');

      -- afisare actori
      exista_actori := FALSE;
      FOR actor in actori(serial.series_id) LOOP
        exista_actori := TRUE;
        DBMS_OUTPUT.PUT(actor.first_name || ' ' || actor.last_name || ' - ');
      END LOOP;
    END LOOP;
END;

```

```

-- afisare personajul jucat de actor
SELECT c.first_name, c.last_name INTO personaj
FROM characters c JOIN playing p USING(character_id)
WHERE actor_id = actor.actor_id;

DBMS_OUTPUT.PUT(personaj.prenume || ' ' || personaj.num);
DBMS_OUTPUT.PUT_LINE("");
END LOOP;

IF NOT exista_actors THEN
    -- nu exista actors
    DBMS_OUTPUT.PUT_LINE('Nu exista actors');
END IF;

DBMS_OUTPUT.PUT_LINE("");
END LOOP;
END;
/

```

```

-----
--- SUPERNATURAL ---
--- Producatori: Eric Kripke ---
-----
Jared Padalecki - Sam Winchester
Jensen Ackles - Dean Winchester
Misha Collins - Castiel
Mark Sheppard - Crowley
Alexander Calvert - Jack
Rob Benedict - God

-----
--- GOSSIP GIRL ---
--- Producatori: Stephanie Savage, Josh Schwartz ---
-----
Blake Lively - Serena van der Woodsen
Leighton Master - Blair Waldorf
Penn Badgley - Dan Humphrey
Ed Westwick - Chuck Bass
Chace Crawford - Nate Archibald

-----
--- THE ORIGINALS ---
--- Producatori: Julie Plec ---
-----
Joseph Morgan - Klaus Mikaelson
Daniel Gillies - Elijah Mikaelson
Claire Holt - Rebekah Mikaelson
Riley Voelkel - Freya Mikaelson
Nathaniel Buzolic - Kol Mikaelson

```

10. Definirea unui trigger LMD la nivel de comanda

Modificarea tabelului series nu se poate fi realizata decât de userul c##claudia, în intervalul Luni - Vineri între orele 8 - 20.

```
CREATE OR REPLACE TRIGGER modificare_serial
  BEFORE INSERT OR UPDATE OR DELETE ON series
BEGIN
  IF UPPER(SYS.LOGIN_USER) <> 'C##CLAUDIA' THEN
    RAISE_APPLICATION_ERROR(-20008, 'Nu aveti dreptul de a modifica acest tabel');
  ELSIF TO_CHAR(SYSTIMESTAMP, 'D') IN (1, 7) THEN
    RAISE_APPLICATION_ERROR(-20009, 'Nu se poate modifica tabelul in zilele de weekend');
  ELSIF TO_CHAR(SYSTIMESTAMP, 'HH24') NOT BETWEEN 8 AND 20 THEN
    RAISE_APPLICATION_ERROR(-20010, 'Nu se poate modifica tabelul in afara intervalului 8 - 20');
  END IF;
END;
```

```
-- 20:59 => nu merge
INSERT INTO series
VALUES (8, 'Nikita', 'A rogue assassin returns to take down the secret organization that tra
       7.7, TO_DATE('9-Sep-2010', 'DD MONTH YYYY'), TO_DATE('27-Dec-2013', 'DD MONTH YYYY'))
```

Script Output x

Task completed in 0.201 seconds

```
Error starting at line : 20 in command -
INSERT INTO series
VALUES (8, 'Nikita', 'A rogue assassin returns to take down the secret organization that trained
       7.7, TO_DATE('9-Sep-2010', 'DD MONTH YYYY'), TO_DATE('27-Dec-2013', 'DD MONTH YYYY'), ger
Error report -
ORA-20010: Nu se poate modifica tabelul in afara intervalului 8 - 20
ORA-06512: at "C##CLAUDIA.MODIFICARE_SERIAL", line 7
ORA-04088: error during execution of trigger 'C##CLAUDIA.MODIFICARE_SERIAL'
```

```
-- sambata => nu merge
INSERT INTO series
VALUES (8, 'Nikita', 'A rogue assassin returns to take down the secret organization that tra
       7.7, TO_DATE('9-Sep-2010', 'DD MONTH YYYY'), TO_DATE('27-Dec-2013', 'DD MONTH YYYY'))
```

Script Output x

Query Result x

Task completed in 0.159 seconds

```
INSERT INTO series
VALUES (8, 'Nikita', 'A rogue assassin returns to take down the secret organization that trained
       7.7, TO_DATE('9-Sep-2010', 'DD MONTH YYYY'), TO_DATE('27-Dec-2013', 'DD MONTH YYYY'), ger
Error report -
ORA-20009: Nu se poate modifica tabelul in zilele de weekend
ORA-06512: at "C##CLAUDIA.MODIFICARE_SERIAL", line 5
ORA-04088: error during execution of trigger 'C##CLAUDIA.MODIFICARE_SERIAL'
```

11. Definirea unui trigger LMD la nivel de linie

Un sezon nu poate sa aibă mai mult de 3 episoade.

```
-- creare copie a tabelului episodes
CREATE TABLE episodes_cpy
AS SELECT * FROM episodes;

-- functie care returneaza nr de episoade ale unui sezon
CREATE OR REPLACE FUNCTION nr_episoade_sezon
(id_sez seasons.season_id%TYPE)
RETURN NUMBER IS
    nr_ep NUMBER(1);
BEGIN
    SELECT COUNT(*) INTO nr_ep
    FROM episodes_cpy
    WHERE season_id = id_sez;

    RETURN nr_ep;
END;
/

-- creare trigger
CREATE OR REPLACE TRIGGER modificare_episodes
BEFORE INSERT OR UPDATE ON episodes
FOR EACH ROW
BEGIN
    IF nr_episoade_sezon(:NEW.season_id) = 3 THEN
        RAISE_APPLICATION_ERROR(-20011, 'Un sezon nu poate avea mai mult de 3 episoade');
    END IF;
END;
/

-- trigger care actualizeaza tabela episodes_cpy
CREATE OR REPLACE TRIGGER actualizare_episodes_cpy
AFTER INSERT OR UPDATE OR DELETE ON episodes
FOR EACH ROW
BEGIN
    IF INSERTING THEN
        INSERT INTO episodes_cpy
        VALUES (:NEW.episode_id, :NEW.episode_number, :NEW.title, :NEW.description,
```

```

        :NEW.minutes, :NEW.airing_date, :NEW.rating, :NEW.season_id);
ELSIF UPDATING THEN
    UPDATE episodes_cpy
    SET episode_number = :NEW.episode_number,
        title = :NEW.title,
        description = :NEW.description,
        minutes = :NEW.minutes,
        airing_date = :NEW.airing_date,
        rating = :NEW.rating,
        season_id = :NEW.season_id
    WHERE episode_id = :OLD.episode_id;
ELSE
    DELETE FROM episodes_cpy
    WHERE episode_id = :OLD.episode_id;
END IF;
END;
/

```

```

-- sezonul are doar 2 episoade => merge
INSERT INTO episodes
VALUES (19, 19, 'No More Heartbreaks', 'Everyone joins together in an attempt to save Cami'
      41, '29-APR-2016', 9.3, 6);

```

Script Output x

Task completed in 0.238 seconds

1 row inserted.

```

-- sezonul are deja 3 episoade => nu merge
INSERT INTO episodes
VALUES (20, 11, 'Wild at Heart', 'Elijah learns that Aya might have knowlege about an elusive wea
      42, '05-FEB-2016', 8.6, 6);

```

Script Output x

Task completed in 0.285 seconds

Error starting at line : 105 in command -

```

INSERT INTO episodes
VALUES (20, 11, 'Wild at Heart', 'Elijah learns that Aya might have knowlege about an elusive wea
      42, '05-FEB-2016', 8.6, 6)
Error report -
ORA-20011: Un sezon nu poate avea mai mult de 3 episoade
ORA-06512: at "C##CLAUDIA.MODIFICARE_EPISODES", line 3
ORA-04088: error during execution of trigger 'C##CLAUDIA.MODIFICARE_EPISODES'

```



```
-- sezonul are deja 3 episoade => nu merge
UPDATE episodes
SET season_id = 6
WHERE episode_id = 7;
```

Script Output x | Task completed in 0.073 seconds

```
Error starting at line : 110 in command -
UPDATE episodes
SET season_id = 6
WHERE episode_id = 7
Error report -
ORA-20011: Un sezon nu poate avea mai mult de 3 episoade
ORA-06512: at "C##CLAUDIA.MODIFICARE_EPISODES", line 3
ORA-04088: error during execution of trigger 'C##CLAUDIA.MODIFICARE_EPISODES'
```

```
-- sezonul are doar 2 episoade, deci nu merg inserate alte 3 => nu merge
CREATE SEQUENCE sec_episodes
START WITH 20
INCREMENT BY 1;

BEGIN
  FOR i IN 1..5 LOOP
    INSERT INTO episodes
    VALUES (sec_episodes.NEXTVAL, 2, 'Red Queen', 'Octavia is forced to take guidance fr
    42, '01-MAY-2018', 8.8, 10);
  END LOOP;
END;
```

Script Output x | Task completed in 0.684 seconds

```
FOR i IN 1..5 LOOP
  INSERT INTO episodes
  VALUES (sec_episodes.NEXTVAL, 2, 'Red Queen', 'Octavia is forced to take guidance from ar
  42, '01-MAY-2018', 8.8, 10);
END LOOP;
END;
Error report -
ORA-20011: Un sezon nu poate avea mai mult de 3 episoade
ORA-06512: at "C##CLAUDIA.MODIFICARE_EPISODES", line 3
ORA-04088: error during execution of trigger 'C##CLAUDIA.MODIFICARE_EPISODES'
ORA-06512: at line 3
```

12. Definirea unui trigger LDD

Pentru fiecare comanda LDD efectuata sa se insereze in tabela istoric_comenzi numele comenzii, obiectul asupra căreia a fost efectuată, data efectuării și utilizatorul ce a efectuat comanda.

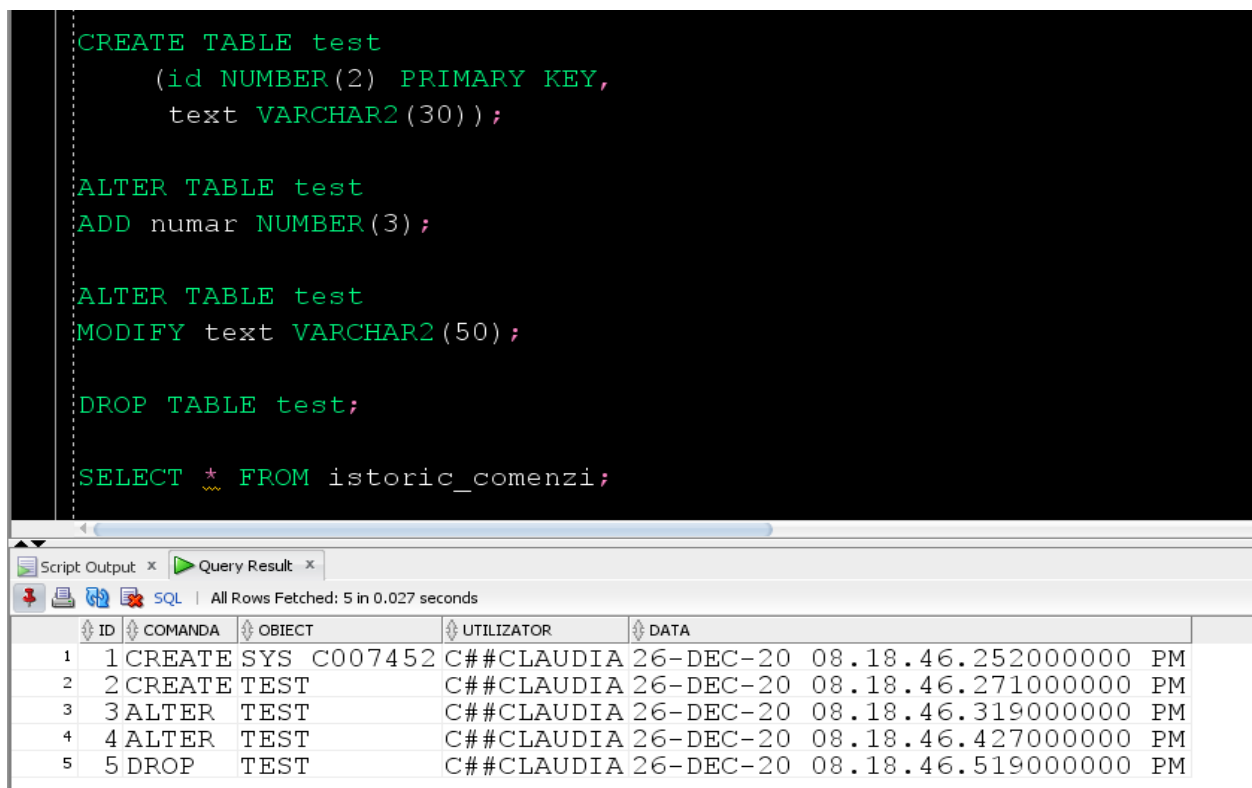
```

-- creare tabel istoric_comenzi
CREATE TABLE istoric_comenzi
  (id NUMBER(3) PRIMARY KEY,
   comanda VARCHAR2(20),
   obiect VARCHAR2(30),
   utilizator VARCHAR2(30),
   data TIMESTAMP);

-- creare secventa
CREATE SEQUENCE sec_istoric_comenzi
START WITH 1
INCREMENT BY 1;

-- creare trigger
CREATE OR REPLACE TRIGGER comenzi_ldd
  AFTER CREATE OR ALTER OR DROP ON SCHEMA
BEGIN
  INSERT INTO istoric_comenzi
  VALUES          (sec_istoric_comenzi.NEXTVAL,          SYS.SYSEVENT,
SYS.DICTIONARY_OBJ_NAME, SYS.LOGIN_USER, SYSTIMESTAMP);
END;
/

```



The screenshot displays a SQL IDE interface. The top pane shows a script with the following SQL commands:

```

CREATE TABLE test
  (id NUMBER(2) PRIMARY KEY,
   text VARCHAR2(30));

ALTER TABLE test
ADD numar NUMBER(3);

ALTER TABLE test
MODIFY text VARCHAR2(50);

DROP TABLE test;

SELECT * FROM istoric_comenzi;

```

The bottom pane shows the 'Query Result' tab, indicating 'All Rows Fetched: 5 in 0.027 seconds'. The result is a table with 5 rows and 7 columns: ID, COMANDA, OBIECT, UTILIZATOR, and DATA (split into two columns for date and time).

ID	COMANDA	OBIECT	UTILIZATOR	DATA	
1	1 CREATE	SYS C007452	C##CLAUDIA	26-DEC-20 08.18.46.252000000	PM
2	2 CREATE	TEST	C##CLAUDIA	26-DEC-20 08.18.46.271000000	PM
3	3 ALTER	TEST	C##CLAUDIA	26-DEC-20 08.18.46.319000000	PM
4	4 ALTER	TEST	C##CLAUDIA	26-DEC-20 08.18.46.427000000	PM
5	5 DROP	TEST	C##CLAUDIA	26-DEC-20 08.18.46.519000000	PM

13. Definirea unui pachet care sa conțină toate obiectele definite în cadrul proiectului

```
CREATE OR REPLACE PACKAGE pachet_1 AS
    PROCEDURE modificare_categorii
        (serial series.title%TYPE,
         categ1 VARCHAR2,
         optiune VARCHAR2,
         categ2 VARCHAR2 := NULL);

    PROCEDURE afisare_episoade
        (serial series.title%TYPE);

    FUNCTION nr_episoade
        (prenume actors.first_name%TYPE := NULL,
         nume actors.last_name%TYPE := NULL,
         inceput DATE,
         sfarsit DATE)
    RETURN NUMBER;

    PROCEDURE afisare_seriale;
END pachet_1;
/

CREATE OR REPLACE PACKAGE BODY pachet_1 AS
    PROCEDURE modificare_categorii
        (serial series.title%TYPE,
         categ1 VARCHAR2,
         optiune VARCHAR2,
         categ2 VARCHAR2 := NULL)
    AS
        categorii genres;
        i INTEGER;
    BEGIN
        -- obtinere lista categorii pentru seriaialul dat
        SELECT genre INTO categorii
        FROM series
        WHERE title = INITCAP(serial);

        IF UPPER(optiune) = 'INSERTING' THEN
            IF categ1 IS NULL THEN
```

```
        RAISE_APPLICATION_ERROR(-20001, 'Categoria nou introdusa nu poate sa fie
NULL');
    ELSE
        -- adaugarea unei categorii noi
        categorii.extend();
        categorii(categorii.last) := INITCAP(categ1);
    END IF;
ELSIF UPPER(optiune) = 'DELETING' THEN
    IF categ1 IS NULL THEN
        RAISE_APPLICATION_ERROR(-20001, 'Categoria de sters nu poate sa fie NULL');
    ELSE
        -- determinarea pozitiei categoriei ce trebuie stersa
        i := categorii.FIRST;
        WHILE i <= categorii.LAST AND categorii(i) <> INITCAP(categ1) LOOP
            i := categorii.NEXT(i);
        END LOOP;

        IF i IS NOT NULL THEN
            -- stergerea categoriei
            categorii.DELETE(i);
        ELSE
            RAISE_APPLICATION_ERROR(-20002, 'Nu exista categoria introdusa');
        END IF;
    END IF;
ELSIF UPPER(optiune) = 'UPDATING' THEN
    IF categ1 IS NULL THEN
        RAISE_APPLICATION_ERROR(-20001, 'Categoria de actualizat nu poate sa fie
NULL');
    ELSE
        IF categ2 IS NULL THEN
            RAISE_APPLICATION_ERROR(-20001, 'Categoria nou introdusa nu poate sa fie
NULL');
        ELSE
            -- determinarea pozitiei categoriei ce trebuie modificata
            i := categorii.FIRST;
            WHILE i <= categorii.LAST AND categorii(i) <> INITCAP(categ1) LOOP
                i := categorii.NEXT(i);
            END LOOP;

            IF i IS NOT NULL THEN
                -- odificarea categoriei
                categorii(i) := INITCAP(categ2);
```

```

        ELSE
            RAISE_APPLICATION_ERROR(-20002, 'Nu exista categoria introdusa');
        END IF;
    END IF;
END IF;
ELSE
    RAISE_APPLICATION_ERROR(-20003, 'Optiunea introdusa este gresita');
END IF;

-- actualizare lista categorii
UPDATE series
SET genre = categorii
WHERE title = serial;

DBMS_OUTPUT.PUT_LINE('Lista de categorii a fost actualizata cu succes');
EXCEPTION
    WHEN no_data_found THEN
        RAISE_APPLICATION_ERROR(-20004, 'Nu exista serial cu numele dat');
    WHEN too_many_rows THEN
        RAISE_APPLICATION_ERROR(-20005, 'Exista mai multe seriale cu acest nume');
END;

PROCEDURE afisare_episoade
(serial series.title%TYPE)
AS
    TYPE ref_cursor IS REF CURSOR;
    CURSOR sezoane (id_serial NUMBER) IS
        SELECT season_number, starting_date, ending_date,
            CURSOR (SELECT episode_number, title, description, rating
                FROM episodes e
                WHERE e.season_id = s.season_id)
        FROM seasons s
        WHERE series_id = id_serial;
    episoade ref_cursor;
    id_serial series.series_id%TYPE;
    numar_sez seasons.season_number%TYPE;
    inceput_sez seasons.starting_date%TYPE;
    sfarsit_sez seasons.ending_date%TYPE;
    TYPE ep IS RECORD (numar episodes.episode_number%TYPE,
        titlu episodes.title%TYPE,
        descriere episodes.description%TYPE,

```

```
        rating episodes.rating%TYPE);
episod ep;
exista_sezoane BOOLEAN := FALSE;
exista_episoade BOOLEAN;
BEGIN
    -- determinare id serial
    SELECT series_id INTO id_serial
    FROM series
    WHERE title = serial;

    OPEN sezoane(id_serial);
    LOOP
        FETCH sezoane INTO numar_sez, inceput_sez, sfarsit_sez, episoade;
        EXIT WHEN sezoane%NOTFOUND;
        exista_sezoane := TRUE;

        -- afisare sezon
        DBMS_OUTPUT.PUT_LINE('-----');
        DBMS_OUTPUT.PUT('SEZONUL ' || numar_sez || ': ' || inceput_sez || ' - ');

        IF sfarsit_sez IS NULL THEN
            -- sezonul se afla in derulare
            DBMS_OUTPUT.PUT_LINE('prezent');
        ELSE
            DBMS_OUTPUT.PUT_LINE(sfarsit_sez);
        END IF;

        -- afisare episoade
        exista_episoade := FALSE;

        LOOP
            FETCH episoade INTO episod;
            EXIT WHEN episoade%NOTFOUND;

            exista_episoade := TRUE;

            DBMS_OUTPUT.PUT_LINE(episod.numar || ' ' || episod.titlu || ' - ' || episod.rating);
            DBMS_OUTPUT.PUT_LINE('Synopsis: ' || episod.descriere);
            DBMS_OUTPUT.PUT_LINE("");
        END LOOP;

        IF NOT exista_episoade THEN
```

```
        DBMS_OUTPUT.PUT_LINE('Nu exista episoade pentru acest sezon');
    END IF;

    DBMS_OUTPUT.PUT_LINE('-----');
    DBMS_OUTPUT.PUT_LINE("");
END LOOP;
CLOSE sezoane;

IF NOT exista_sezoane THEN
    DBMS_OUTPUT.PUT_LINE('Nu exista sezoane pentru acest serial');
END IF;
EXCEPTION
    WHEN no_data_found THEN
        RAISE_APPLICATION_ERROR(-20004, 'Nu exista serial cu numele dat');
    WHEN too_many_rows THEN
        RAISE_APPLICATION_ERROR(-20005, 'Exista mai multe seriale cu acest nume');
END;

FUNCTION nr_episoade
(prenume actors.first_name%TYPE := NULL,
 nume actors.last_name%TYPE := NULL,
 inceput DATE,
 sfarsit DATE)
RETURN NUMBER IS
    nr_ep NUMBER(3);
    id_actor actors.actor_id%TYPE;
BEGIN
    IF inceput > sfarsit THEN
        RAISE_APPLICATION_ERROR(-20006, 'Data de inceput trebuie sa fie mai mica decat
data de sfarsit');
        RETURN -1;
    END IF;

    IF prenume IS NULL AND nume IS NULL THEN
        RAISE_APPLICATION_ERROR(-20007, 'Nu poate sa fie si numele si prenumele
NULL');
        RETURN -1;
    END IF;

    -- determinarea id-ului actorului dat
    -- (acest pas se face separat ca sa se poata arunca exceptie in cazul in care
```

```
-- nu exista actorulul sau exista mai multi acotri cu acest nume)
IF nume IS NOT NULL AND prenume IS NOT NULL THEN
    -- numele si prenumele nusunt NULL
    SELECT actor_id INTO id_actor
    FROM actors
    WHERE first_name = prenume
        AND last_name = nume;
ELSIF nume IS NULL AND prenume IS NOT NULL THEN
    -- prenumele nu este NULL
    SELECT actor_id INTO id_actor
    FROM actors
    WHERE first_name = prenume;
ELSE
    -- numele nu este NULL
    SELECT actor_id INTO id_actor
    FROM actors
    WHERE last_name = nume;
END IF;

SELECT COUNT(*) INTO nr_ep
FROM playing p JOIN characters ch ON (p.character_id = ch.character_id)
    JOIN appearing_in ap ON (ch.character_id = ap.character_id)
    JOIN episodes e ON (ap.episode_id = e.episode_id)
WHERE p.actor_id = id_actor
    AND (p.starting_date <= sfarsit AND p.ending_date >= inceput)
    AND e.airing_date BETWEEN inceput AND sfarsit;

RETURN nr_ep;
EXCEPTION
    WHEN no_data_found THEN
        RAISE_APPLICATION_ERROR(-20004, 'Nu exista actor cu numele dat');
        RETURN -1;
    WHEN too_many_rows THEN
        RAISE_APPLICATION_ERROR(-20005, 'Exista mai multi actori cu acest nume');
        RETURN -1;
END;

PROCEDURE afisare_seriale AS
    TYPE pers IS RECORD (prenume producers.first_name%TYPE,
        nume producers.last_name%TYPE);
```



```

TYPE prod IS TABLE OF pers;
v_producatori prod;
CURSOR actori (id_serial NUMBER) IS
    SELECT a.actor_id, a.first_name, a.last_name
    FROM actors a JOIN playing p ON(a.actor_id = p.actor_id)
    WHERE series_id = id_serial;
personaj pers;
i INTEGER;
exista_actors BOOLEAN;
BEGIN
    FOR serial IN (SELECT series_id, title
                    FROM series)
        LOOP
            -- afisare serial
            DBMS_OUTPUT.PUT_LINE('-----');
            DBMS_OUTPUT.PUT_LINE('--- ' || UPPER(serial.title) || ' ---');

            -- afisare producatori
            DBMS_OUTPUT.PUT('--- Producatori: ');

            SELECT p.first_name, p.last_name BULK COLLECT INTO v_producatori
            FROM producers p JOIN produced_by ps ON (p.producer_id = ps.producer_id)
            WHERE series_id = serial.series_id;

            IF v_producatori.count() = 0 THEN
                -- nu exista producatori
                DBMS_OUTPUT.PUT('nu exista producatori');
            ELSE
                i := v_producatori.FIRST;
                WHILE i <= v_producatori.LAST LOOP
                    DBMS_OUTPUT.PUT(v_producatori(i).prename || ' ' || v_producatori(i).nume);
                    IF i <> v_producatori.LAST THEN
                        DBMS_OUTPUT.PUT(', ');
                    END IF;

                    i := v_producatori.NEXT(i);
                END LOOP;
            END IF;

            DBMS_OUTPUT.PUT(' ---');
            DBMS_OUTPUT.PUT_LINE('');
            DBMS_OUTPUT.PUT_LINE('-----');
        END LOOP;
END IF;

```

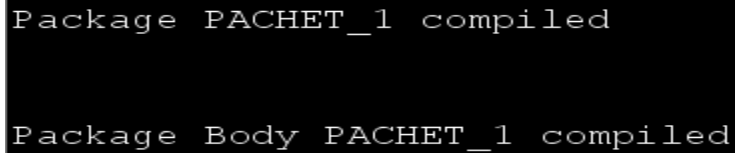
```
-- afisare actori
exista_actori := FALSE;
FOR actor in actori(serial.series_id) LOOP
    exista_actori := TRUE;
    DBMS_OUTPUT.PUT(actor.first_name || ' ' || actor.last_name || ' - ');

    -- afisare personajul jucat de actor
    SELECT c.first_name, c.last_name INTO personaj
    FROM characters c JOIN playing p USING(character_id)
    WHERE actor_id = actor.actor_id;

    DBMS_OUTPUT.PUT(personaj.prenume || ' ' || personaj.num);
    DBMS_OUTPUT.PUT_LINE("");
END LOOP;

IF NOT exista_actori THEN
    -- nu exista actori
    DBMS_OUTPUT.PUT_LINE('Nu exista actori');
END IF;

DBMS_OUTPUT.PUT_LINE("");
END LOOP;
END;
END pachet_1;
/
```



```
Package PACHET_1 compiled

Package Body PACHET_1 compiled
```