

Scheda Sintetica

La scheda sintetica va compilata ed inviata al docente, in forma elettronica, *prima* di sostenere l'esame, unitamente alla documentazione e al codice. Una sua copia va inoltre allegata alla documentazione del progetto.

Nome del progetto:

Autori:

Nome	Cognome	Matricola	Ruolo nello sviluppo del progetto
Andrea	Segnalini	243859	Documentazione, Database, Interrogazioni
Gabriele	Di Egidio	260035	Leader, Database, Interrogazioni

Data di consegna del progetto:

07/07/2022

Laboratorio basi di dati

GitHub: <https://github.com/Caprielos/Lbd.git>

Andrea Segnalini 248859 andrea.segnalini@student.univaq.it

Gabriele Di Egidio 260035 gabriele.diegidio@student.univaq.it

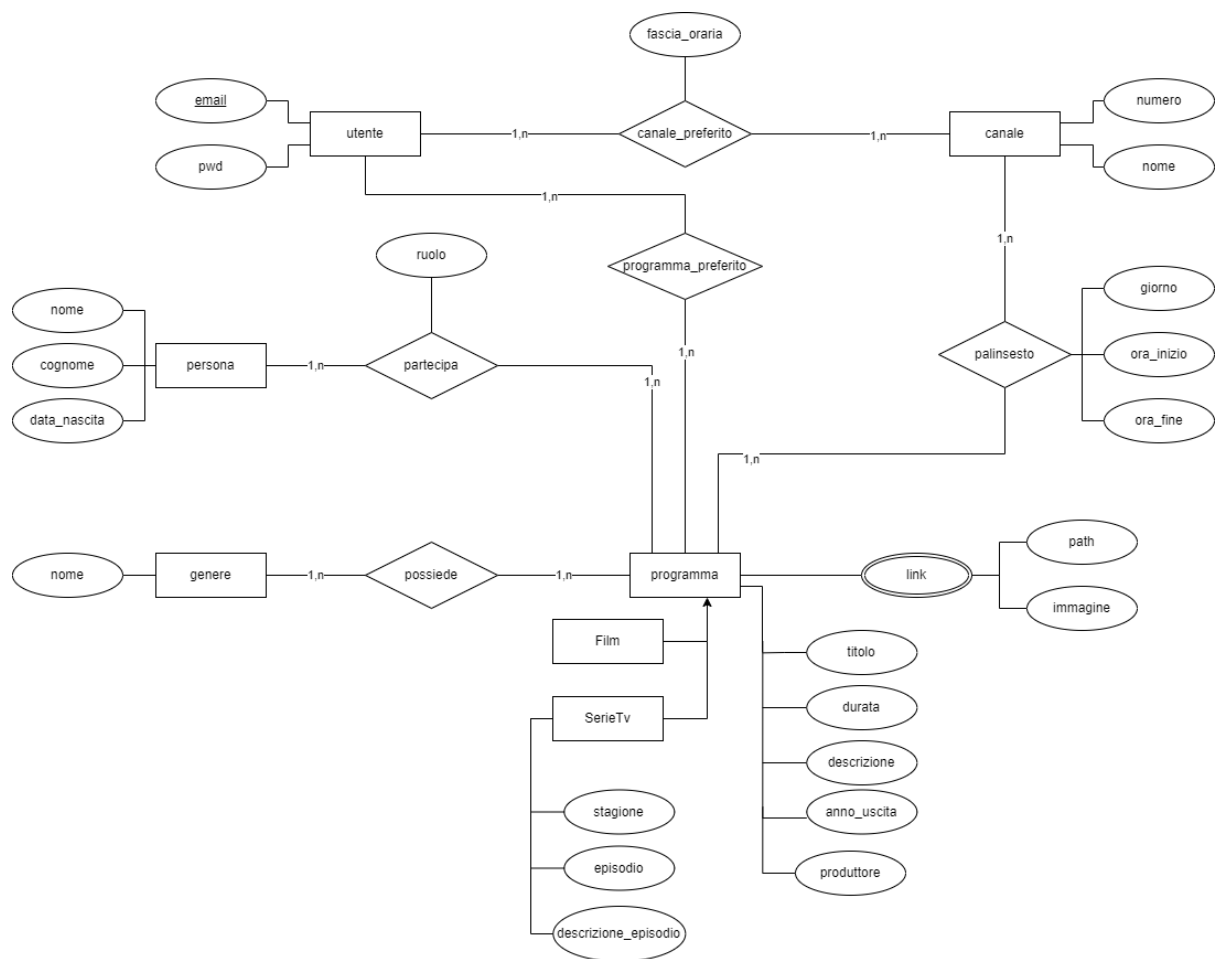
Entità

- **Utente**
Utenti posso visitare i canali (sinonimi: utilizzatore, cliente consumatore)
- **Canale**
Il canale contiene i palinsesti di ogni giorno (sinonimi: emittente, stazione televisiva)
- **Programma**
Contiene l'episodio o il film (sinonimi: cartellone, spettacolo)
- **Persone**
Persone presenti nel dato programma, registi, attori e altro (sinonimi: troupe)
- **Genere** (sinonimi: non pervenuto)
Lista di generi che può avere un programma
- **Film** (sinonimi: pellicola, lungometraggio)
Collezioni di un film
- **Serie Tv** (sinonimi: film a episodi)
Collezioni di film a episodi

Relazioni

- **Palinsesto**
Lista oraria di programmi in proiezione nel medesimo giorno
- **Partecipa**
Troupe del film
- **Possiede**
Generi del singolo programma
- **Canale preferito**
Lista canali preferiti del singolo utente
- **Programma preferito**
Lista programma preferiti dal singolo utente

ER



Vincoli

- **Giorni successivi**
Pubblicazione massima dei sette giorni successivi alla data odierna
- **Limite Giorni Passati**
Si ha un limite per visualizzare i programmi dei giorni passati
- **Ripetizioni dei programmi**
Non è possibile inserire duplicati dello stesso programma
- **Unique utente**
La mail dell'utente forma un attributo di tipo unique, useremo una regex per validare che la mail inserita sia conforme alla regex.
- **Unique canale**
Il titolo del canale e il suo numero formano un attributo unique
- **Unique genere**
Il nome forma un attributo unique
- **Unique programma**
Gli attributi: titolo, anno uscita, produttore per il programma formano un attributo di tipo unique
- **Unique programma**
Gli attributi: titolo, anno uscita, produttore, stagione, episodio per il programma formano un attributo di tipo unique

- **Durata programmi palinsesto**

La somma della durata di un programma non può eccedere alle ore di fine del palinsesto, inoltre non possono esserci due o più palinsesti nello stesso orario e canale

- **Ora inizio palinsesto**

L'inizio di un programma deve essere maggiore o uguale dell'ora di fine del precedente palinsesto

- **Persona Unique**

Gli attributi: nome, cognome, data di nascita per Persona formano un attributo unique

Assunzioni

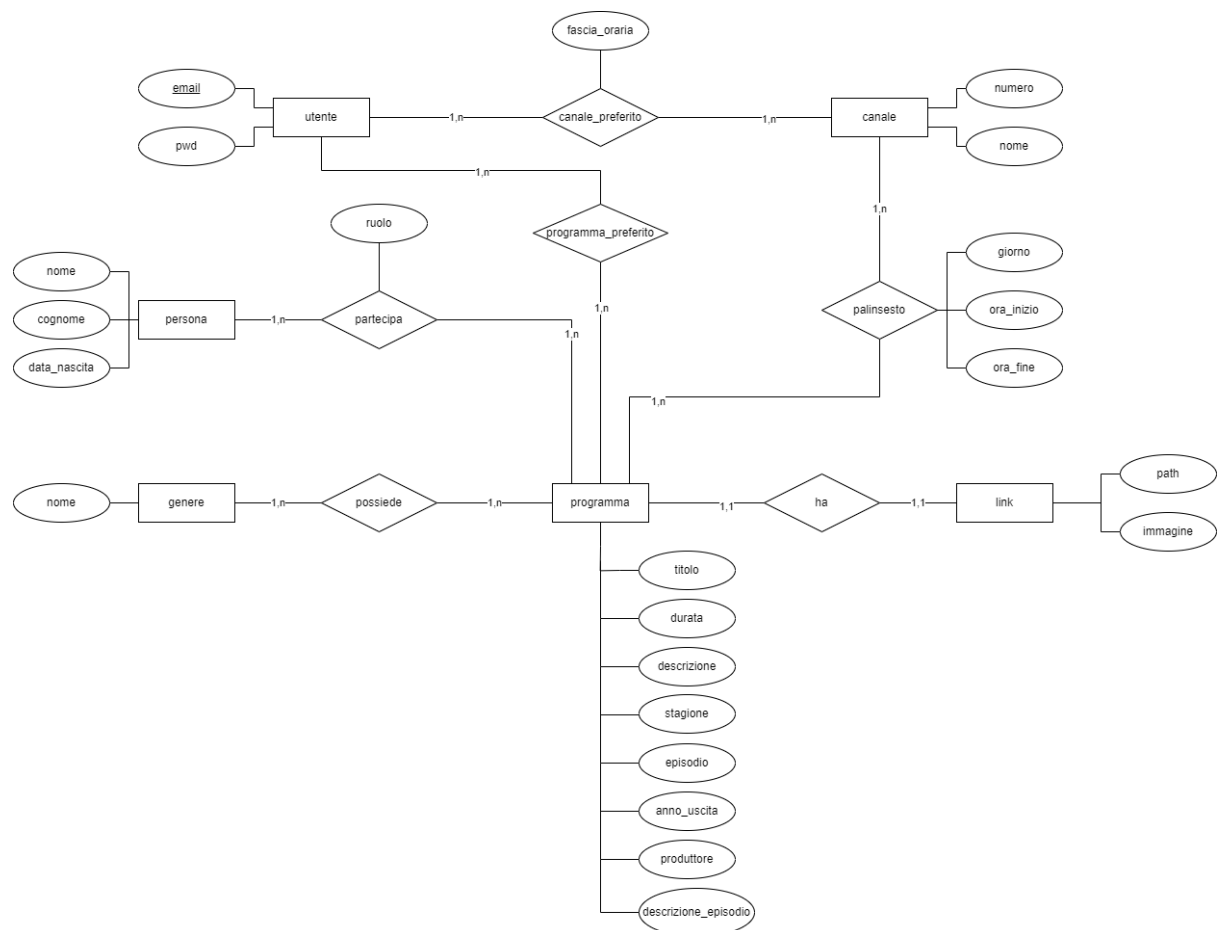
- **Pubblicità**

Pubblicità fanno parte della durata del palinsesto, non verrà gestita dal database

- **Amministratore**

Inserisce i programmi, decide l'ordine del palinsesto

Ristrutturazione



- **Programma**

Abbiamo scelto di accorpare Film e Serie Tv dentro Programma, i due si differenziano tra loro tramite gli attributi episodio, stagione e descrizione episodio, nei film vengono impostati a null

Modello relazionale

Interrogazioni

1. Procedura₁: Registrazione utente

Query per inserimento della utente, quando si inserisce una mail, controlliamo tramite una Regex la validità della email. In caso contrario solleva un errore

```
3 DELIMITER $
4
5 • -- 1 > [Registrazione (inserimento dei dati del profilo) di un utente.] < --
6 CREATE PROCEDURE `query_1`(IN email_param VARCHAR(64), IN pwd_param VARCHAR(32))
7 BEGIN
8     IF (SELECT email_param NOT REGEXP '[a-zA-Z0-9._-].*@[a-zA-Z0-9._-].*\.[a-zA-Z0-9].')
9     THEN
10         SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'email inserita non corretta';
11     ELSE
12         INSERT INTO `guida_tv`.utente(`email`, `pwd`) VALUES (email_param, pwd_param);
13     END IF;
14 END $
15
16 DELIMITER $
```

	id	email	pwd	orario_email
▶	1	diegidiogabriele@gmail.com	1234	Mattina
	2	andreasegnalini@gmail.com	1234	Sera
*	NULL	NULL	NULL	NULL

2. Procedura_2: Inserimento Film/Episodio: Controlliamo anche se o il film o l'episodio è duplicato, se è duplicato lanciamo un errore

```

24 -- 2A > [Inserimento di un film.] --
25
26 CREATE PROCEDURE `guida_tv`.`query_2A` (IN titolo_param VARCHAR(64),
27                                         IN durata_param TIME,
28                                         IN descrizione_param VARCHAR(128),
29                                         IN anno_uscita_param DATE,
30                                         IN path_param VARCHAR(255),
31                                         IN produttore_param VARCHAR(50),
32                                         IN immagine_param LONGBLOB)
33
34 BEGIN
35
36     IF EXISTS (SELECT u.titolo FROM programma u WHERE u.titolo = titolo_param AND u.anno_uscita = anno_uscita_param AND u.prodotto = produttore_param)
37     THEN
38         SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'Questo film è già stato inserito';
39     ELSE
40         INSERT INTO `guida_tv`.`programma` ('titolo', 'durata', 'descrizione', 'anno_uscita', 'stagione', 'episodio', 'produttore', 'descrizione_episodio',
41                                             'path', 'immagine')
42         VALUES (titolo_param, durata_param, descrizione_param, anno_uscita_param, NULL, NULL, produttore_param, NULL, path_param, immagine_param);
43     END IF;
44
45 END $
46
47 -- 2B > [Inserimento di un episodio di una serie.] < --
48
49 CREATE PROCEDURE `query_2B` (IN titolo_param VARCHAR(64),
50                              IN durata_param TIME,
51                              IN descrizione_param VARCHAR(128),
52                              IN anno_uscita_param DATE,
53                              IN stagione_param TINYINT,
54                              IN episodio_param MEDIUMINT,
55                              IN produttore_param VARCHAR(50),
56                              IN descrizione_episodio_param VARCHAR(128),
57                              IN path_param VARCHAR(255),
58                              IN immagine_param LONGBLOB)
59
60 BEGIN
61
62     IF EXISTS (SELECT u.titolo FROM programma u WHERE u.titolo = titolo_param AND u.anno_uscita = anno_uscita_param AND u.prodotto = produttore_param
63               AND u.stagione = stagione_param AND u.episodio = episodio_param)
64     THEN
65         SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'Questo episodio è già stato inserito';
66     ELSE
67         INSERT INTO `guida_tv`.`programma` ('titolo', 'durata', 'descrizione', 'anno_uscita', 'stagione', 'episodio', 'produttore', 'descrizione_episodio',
68                                             'path', 'immagine')
69         VALUES (titolo_param, durata_param, descrizione_param, anno_uscita_param, stagione_param, episodio_param, produttore_param, descrizione_episodio_param,
70                 path_param, immagine_param);
71     END IF;
72
73 END $
74
75 DELIMITER $

```

[illegible]

3. Procedura_3: Palinsesto giornaliero dato un canale: Restituisce il palinsesto della data odierna dato un canale

```
4
5 • -- 3 > [Generazione del palinsesto odierno di un canale.] < --
6 CREATE PROCEDURE `guida_tv`.`query_3` (IN nome_canale_param VARCHAR(64))
7
8 BEGIN
9
10     SELECT DISTINCT pal.ora_inizio,
11                     prog.titolo,
12                     pal.ora_fine,
13                     gen.nome
14 FROM `guida_tv`.palinsesto pal
15
16 JOIN `guida_tv`.canale can ON pal.id_canale = ( SELECT c.id FROM canale c WHERE c.nome = nome_canale_param )
17 JOIN `guida_tv`.possiede pos ON pos.id_programma = pal.id_programma
18 JOIN `guida_tv`.genere gen ON pos.id_genere = gen.id
19 JOIN `guida_tv`.programma prog ON prog.id = pal.id_programma
20 WHERE pal.giorno = curdate() AND can.nome = nome_canale_param
21 ORDER BY pal.ora_inizio ASC;
22
23 END $
24
```

```
1 • call guida_tv.query_3('Rai 1');
2
```

Result Grid				
Filter Rows:		Export:		
		Wrap Cell Content:		
	ora_inizio	titolo	ora_fine	nome
▶	00:00:00	Rambo 1	02:00:00	Guerra
	02:00:00	Rambo 1	04:00:00	Guerra
	04:00:00	Rocky	06:00:00	Animazione
	11:00:00	Pirati dei caraibi: La maledizione della prima luna	13:30:00	Guerra
	14:00:00	Aladino	18:00:00	Animazione
	14:00:00	Aladino	18:00:00	Avventura

4. **Procedura_4:** Lista canali/date/orari in cui sono trasmessi gli episodi di una certa serie.
Inserita una serie, restituisce la lista della programmazione dei suoi episodi.

```
DELIMITER $
```

```
-- 4 > [Lista dei canali/date/orari in cui sono trasmessi gli episodi di una certa serie] < --
```

```
CREATE PROCEDURE `guida_tv`.`query_4` ( IN nome_serie_param VARCHAR(64))
```

```
BEGIN
```

```
    SELECT can.nome, pal.giorno, pal.ora_inizio, pal.ora_fine, prog.titolo, prog.episodio
```

```
    FROM `guida_tv`.palinsesto pal
```

```
    JOIN `guida_tv`.programma prog ON pal.id_programma = prog.id
```

```
    JOIN `guida_tv`.canale can ON pal.id_canale = can.id
```

```
    WHERE prog.stagione != 0 AND prog.titolo = nome_serie_param ORDER BY pal.giorno ASC;
```

```
    -- WHERE prog.id = (SELECT p.id FROM `guida_tv`.programma p WHERE p.titolo = nome_serie_param);
```

```
END $
```

```
DELIMITER $
```

```
CALL `query_4` ("Games of trones");
```

	nome	giorno	ora_inizio	ora_fine	titolo	episodio
►	Rai 1	2022-06-29	00:00:00	02:00:00	Games of trones	2
	Rai 2	2022-07-01	09:30:00	11:40:00	Games of trones	1

5. Procedura_5: Programmi o Canali maggiormente preferiti dagli utenti

Restituisce i canali o programmi maggiormente preferiti dagli utenti

```
6 • -- 5A > [Lista dei canali maggiormente "preferiti" dagli utenti.] < --
7 CREATE PROCEDURE `guida_tv`.`query_5A` ()
8
9 BEGIN
10
11     SELECT id_canale, COUNT(id_canale) FROM `guida_tv`.`canale_preferito`
12     GROUP BY id_canale
13     HAVING COUNT(id_canale) > 1;
14
15 END $
16
17 -- 5B > [Lista dei programmi maggiormente "preferiti" dagli utenti.] < --
18 • CREATE PROCEDURE `guida_tv`.`query_5B` ()
19
20 BEGIN
21
22     SELECT id_programma, COUNT(id_programma) FROM `guida_tv`.`programma_preferito`
23     GROUP BY id_programma
24     HAVING COUNT(id_programma) > 1;
25
26 END $
```

Risultati query 5A: canali_preferiti

	id_canale	COUNT(id_canale)
▶	1	3
	2	2

Risultati query 5B: programmi_preferiti

	id_programma	COUNT(id_programma)
▶	2	2
	4	3

6. Procedura_6: Eliminazione programma

Elimina il programma richiesto. Eliminando il programma tramite le foreign key, cancelliamo automaticamente tutte le entry nelle altre tabelle.

```
5 • -- 6 > [Eliminazione di un programma televisivo dal database.] < --
6 CREATE PROCEDURE `guida_tv`.`query_6` ( IN titolo_param VARCHAR(64), IN anno_uscita_param DATE, IN produttore_param VARCHAR(50),
7     IN stagione_param TINYINT, IN episodio_param MEDIUMINT)
8
9
10 BEGIN
11     DECLARE id_param INTEGER UNSIGNED;
12
13     IF (stagione_param != NULL AND episodio_param != NULL)
14     THEN
15         -- E UN FILM
16         SET id_param = (SELECT prog.id FROM `guida_tv`.`programma` prog WHERE prog.titolo = titolo_param AND prog.anno_uscita = anno_uscita_param
17                         AND prog.produttore = produttore_param);
18
19         IF NOT EXISTS (SELECT p.id FROM `guida_tv`.`programma` p WHERE id_param =
20                        (SELECT pal.id_programma FROM `guida_tv`.`palinsesto` pal WHERE p.id = pal.id_programma LIMIT 1))
21         THEN
22             DELETE FROM `guida_tv`.`programma` prog WHERE (prog.`id` = id_param);
23             SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'Programma eliminato con successo';
24
25         ELSE
26             SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'Non puoi cancellare una film che e inserita in un palinsesto';
27         END IF;
28     ELSE
29         -- E UNA SERIE TV
30         SET id_param = (SELECT prog.id FROM `guida_tv`.`programma` prog WHERE prog.titolo = titolo_param AND prog.anno_uscita = anno_uscita_param
31                         AND prog.produttore = produttore_param AND prog.stagione = stagione_param AND prog.episodio = episodio_param);
32
33         IF NOT EXISTS (SELECT p.id FROM `guida_tv`.`programma` p WHERE id_param =
34                        (SELECT pal.id_programma FROM `guida_tv`.`palinsesto` pal WHERE p.id = pal.id_programma LIMIT 1))
35         THEN
36             DELETE FROM `guida_tv`.`programma` prog WHERE (prog.`id` = id_param);
37             SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'Serie tv eliminata con successo';
38         ELSE
39             SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'Non puoi cancellare una serie tv che e inserita in un palinsesto';
40         END IF;
41     END IF;
42 END $
43
```

Prima:

id	titolo	durata	desczione	anno_uscita	stagione	episodio	produttore	descrizione_episodio	path	immagine
1	Rambo 1	01:15:00	classico film di guerra	2000-08-17	NULL	NULL	Produttore 1	NULL		BL.00
2	Rocky	01:45:00	box	2000-09-17	NULL	NULL	Produttore 2	NULL		BL.00
3	Aladino	02:00:00	siamo nella sabbia	2003-08-18	NULL	NULL	Produttore 3	NULL		BL.00
4	Io sono leggenda	03:15:00	uno contro tutti	2010-04-27	NULL	NULL	Produttore 4	NULL		BL.00
5	Pirati dei caraibi: La maledizione della prima luna	02:00:00	pirati contro tutti	2000-03-21	NULL	NULL	Produttore 5	NULL		BL.00
6	Pirati dei caraibi: La maledizione del forziere	02:15:00	pirati uno contro tutti	2005-01-02	NULL	NULL	Produttore 5	NULL		BL.00
7	Pirati dei caraibi: Ai confini del mondo	02:45:00	pirati uno contro tutti	2008-06-15	NULL	NULL	Produttore 5	NULL		BL.00
8	Games of trones	00:45:00	serie sui ghiacci	2017-02-01	1	1	Produttore 5	NULL		BL.00
9	Games of trones	00:45:00	serie sui ghiacci	2017-02-01	1	2	Produttore 5	NULL		BL.00
10	Games of trones	00:45:00	serie sui ghiacci	2017-02-01	1	3	Produttore 5	NULL		BL.00
11	Games of trones	00:45:00	serie sui ghiacci	2017-02-01	1	4	Produttore 5	NULL		BL.00
12	Games of trones	00:45:00	serie sui ghiacci	2017-02-01	1	5	Produttore 5	NULL		BL.00
13	Games of trones	00:45:00	serie sui ghiacci	2017-02-01	1	6	Produttore 5	NULL		BL.00
14	Games of trones	00:45:00	serie sui ghiacci	2017-02-01	1	7	Produttore 5	NULL		BL.00
15	Games of trones	00:45:00	serie sui ghiacci	2017-02-01	1	8	Produttore 5	NULL		BL.00
•	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

Call:

```
1 • call guida_tv.query_6('Games of trones', '2017:02:01', 'Produttore 5', 1, 8);
2
```

Dopo:

	id	titolo	durata	descrizione	anno_uscita	stagione	episodio	produttore	descrizione_episodio	path	immagine
▶	1	Rambo 1	01:15:00	classico film di guerra	2000-08-17	NULL	NULL	Produttore 1	NULL		BLOB
	2	Rocky	01:45:00	box	2000-09-17	NULL	NULL	Produttore 2	NULL		BLOB
	3	Aladino	02:00:00	siamo nella sabbia	2003-08-18	NULL	NULL	Produttore 3	NULL		BLOB
	4	Io sono leggenda	03:15:00	uno contro tutti	2010-04-27	NULL	NULL	Produttore 4	NULL		BLOB
	5	Pirati dei caraibi: La maledizione della prima luna	02:00:00	pirati contro tutti	2000-03-21	NULL	NULL	Produttore 5	NULL		BLOB
	6	Pirati dei caraibi: La maledizione del forziere	02:15:00	pirati uno contro tutti	2005-01-02	NULL	NULL	Produttore 5	NULL		BLOB
	7	Pirati dei caraibi: Ai confini del mondo	02:45:00	pirati uno contro tutti	2008-06-15	NULL	NULL	Produttore 5	NULL		BLOB
	8	Games of trones	00:45:00	serie sui ghiacci	2017-02-01	1	1	Produttore 5	NULL		BLOB
	9	Games of trones	00:45:00	serie sui ghiacci	2017-02-01	1	2	Produttore 5	NULL		BLOB
	10	Games of trones	00:45:00	serie sui ghiacci	2017-02-01	1	3	Produttore 5	NULL		BLOB
	11	Games of trones	00:45:00	serie sui ghiacci	2017-02-01	1	4	Produttore 5	NULL		BLOB
	12	Games of trones	00:45:00	serie sui ghiacci	2017-02-01	1	5	Produttore 5	NULL		BLOB
	13	Games of trones	00:45:00	serie sui ghiacci	2017-02-01	1	6	Produttore 5	NULL		BLOB
	14	Games of trones	00:45:00	serie sui ghiacci	2017-02-01	1	7	Produttore 5	NULL		BLOB
*	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

7. Procedura_7: Ricerca film dato un genere

Dato un genere restituisce i film di appartenenza allo stesso

```

5 • -- 7 > [Ricerca dei film di un certo genere in programma nei prossimi sette giorni.] < --
6 CREATE PROCEDURE `guida_tv`.`query_7` ( IN genere_param VARCHAR(64) )
7
8 BEGIN
9
10     SELECT DISTINCT pal.giorno,
11                     prog.titolo,
12                     prog.durata,
13                     prog.descrizione,
14                     prog.anno_uscita,
15                     prog.stagione,
16                     prog.episodio,
17                     prog.produttore,
18                     prog.descrizione_episodio
19
20     FROM `guida_tv`.programma prog
21     JOIN `guida_tv`.palinsesto pal ON pal.id_programma = prog.id
22     JOIN `guida_tv`.possiede aa ON aa.id_programma = prog.id
23     JOIN `guida_tv`.genere gen ON gen.id = aa.id_genere
24     WHERE gen.id = (SELECT g.id FROM `guida_tv`.genere g WHERE g.nome = genere_param)
25     AND pal.giorno >= curdate() AND pal.giorno <= date_add(curdate(), INTERVAL 7 DAY);
26
27 END $
28

```

```

1 • call guida_tv.query_7('Guerra');
2

```

giorno	titolo	durata	descizione	anno_uscita	stagione	episodio	produttore	descrizione_episodio
2022-07-01	Rambo 1	01:15:00	classico film di guerra	2000-08-17	NULL	NULL	Produttore 1	NULL
2022-07-01	Pirati dei caraibi: La maledizione della prima luna	02:00:00	pirati contro tutti	2000-03-21	NULL	NULL	Produttore 5	NULL
2022-07-01	Games of trones	00:45:00	serie sui ghiacci	2017-02-01	1	1	Produttore 5	NULL
2022-06-29	Games of trones	00:45:00	serie sui ghiacci	2017-02-01	1	2	Produttore 5	NULL
2022-06-29	Rambo 1	01:15:00	classico film di guerra	2000-08-17	NULL	NULL	Produttore 1	NULL
2022-06-29	Pirati dei caraibi: La maledizione della prima luna	02:00:00	pirati contro tutti	2000-03-21	NULL	NULL	Produttore 5	NULL

8. Procedura_8: Ricerca film dato un membro dei film

Dato un membro del cast restituisce i film interpretati, nel caso si vuole si può aggiungere il titolo della serie per ricercare gli episodi di appartenenza.

```

5 • -- 8 > [Ricerca dei programmi a cui partecipa a qualsiasi titolo (o con un titolo specificato) una certa persona.] < --
6 CREATE PROCEDURE `guida_tv`.`query_8` ( IN nome_persona_param VARCHAR(64), IN cognome_persona_param VARCHAR(64), IN titolo_programma_param VARCHAR(64))
7
8 BEGIN
9
10 IF (titolo_programma_param != '')
11 THEN
12 -- Trovo i film con nome, cognome e titolo
13 SELECT prog.titolo FROM `guida_tv`.programma prog
14 JOIN `guida_tv`.partecipa part ON part.id_programma = prog.id
15 JOIN `guida_tv`.persona pers ON part.id_persona = pers.id
16 WHERE pers.id = (SELECT DISTINCT p.id FROM `guida_tv`.persona p WHERE p.nome = nome_persona_param AND p.cognome = cognome_persona_param)
17 AND prog.titolo = titolo_programma_param;
18
19 ELSE
20 -- Trovo i film solo con nome e cognome
21 SELECT prog.titolo FROM `guida_tv`.programma prog
22 JOIN `guida_tv`.partecipa part ON part.id_programma = prog.id
23 JOIN `guida_tv`.persona pers ON part.id_persona = pers.id
24 WHERE pers.id = (SELECT DISTINCT p.id FROM `guida_tv`.persona p WHERE p.nome = nome_persona_param AND p.cognome = cognome_persona_param);
25
26 END IF;
27
28 END $

```

```

1 • call guida_tv.query_8('Mauro', 'Zannetti', '');
2

```

titolo
Rambo 1
Pirati dei caraibi: La maledizione del forziere

9. Procedura_9: Numero programmi distinti di un certo canale nella data odierna

Conteggio dei programmi di un palinsesto

```
3 DELIMITER $
4
5 • -- 9 > [Numero programmi distinti trasmessi da ciascuna emittente in un determinato giorno.] < --
6 CREATE PROCEDURE `guida_tv`.`query_9` ( IN giorno_param DATE )
7
8 BEGIN
9
10     SELECT DISTINCT can.nome, COUNT(*)
11     FROM `guida_tv`.`palinsesto` pal
12     JOIN `guida_tv`.`canale` can ON pal.id_canale = can.id
13     JOIN `guida_tv`.`possiede` pos ON pos.id_programma = pal.id_programma
14     JOIN `guida_tv`.`genere` gen ON pos.id_genere = gen.id
15     JOIN `guida_tv`.`programma` prog ON prog.id = pal.id_programma
16     WHERE pal.giorno = (giorno_param)
17     GROUP BY can.nome;
18
19 END $
20
21 DELIMITER $
```

```
1 • call guida_tv.query_9('2022-07-01');
2
```



<		
Result Grid		
Filter Rows:		
Export:		
	nome	COUNT(*)
▶	Rai 1	6
	Rai 2	6

10. Procedura_10: Durata del palinsesto dato un canale

Conteggio della durata del palinsesto giornaliero

```
5 • -- 10 > [Minuti totali di programmazione per un certo canale in un certo giorno.] < --
6 CREATE PROCEDURE `guida_tv`.`query_10` ( IN nome_canale_param VARCHAR(64), IN giorno_param DATE, OUT res TIME )
7
8 BEGIN
9
10 SELECT SEC_TO_TIME( SUM( TIME_TO_SEC( `durata` ) ) )
11 FROM `guida_tv`.`palinsesto` pal
12 JOIN `guida_tv`.`canale` can ON pal.id_canale = ( SELECT c.id FROM canale c WHERE c.nome = nome_canale_param )
13 JOIN `guida_tv`.`programma` prog ON prog.id = pal.id_programma
14 WHERE pal.giorno = giorno_param AND can.nome = nome_canale_param;
15
16 END $
17
18 DELIMITER $
```

```
1 • call guida_tv.query_10('Rai 1', '2022-07-01', @res);
2
```

<	
Result Grid	
Filter Rows: <input type="text"/>	
Export:  Wrap Cell Content: 	
	SEC_TO_TIME(SUM(TIME_TO_SEC(`durata`)))
▶	10:15:00

11. Procedura_11: Generazione mail dei preferiti di un utente

Generazione palinsesto dei preferiti da inserire alla mail

```
6 • -- 11 > [Generazione della email giornaliera per un utente in base alle sue preferenze] < --
7 CREATE PROCEDURE `guida_tv`.`query_11` (IN id_param INTEGER UNSIGNED, OUT email_result TEXT)
8
9 BEGIN
10
11     DECLARE ora_inizio TIME;
12     DECLARE ora_fine TIME;
13     DECLARE canale_nome VARCHAR(64);
14     DECLARE programma_titolo VARCHAR(64);
15     DECLARE count INTEGER UNSIGNED;
16
17     DECLARE exit_loop BOOLEAN DEFAULT FALSE ;
18     DECLARE curs CURSOR FOR SELECT * FROM new_tbl;
19     DECLARE CONTINUE HANDLER FOR NOT FOUND SET exit_loop = TRUE;
20
21     IF (SELECT c_p.fascia_oraria FROM canale_preferito c_p WHERE c_p.id_utente = id_param) = "Mattina"
22     THEN
23         SET ora_inizio = '06:00:00';
24         SET ora_fine = '12:00:00';
25     ELSEIF (SELECT c_p.fascia_oraria FROM canale_preferito c_p WHERE c_p.id_utente = id_param) = "Pomeriggio"
26     THEN
27         SET ora_inizio = '12:00:00';
28         SET ora_fine = '18:00:00';
29     ELSEIF (SELECT c_p.fascia_oraria FROM canale_preferito c_p WHERE c_p.id_utente = id_param) = "Sera"
30     THEN
31         SET ora_inizio = '18:00:00';
32         SET ora_fine = '00:00:00';
33     ELSEIF (SELECT c_p.fascia_oraria FROM canale_preferito c_p WHERE c_p.id_utente = id_param) = "Notte"
34     THEN
35         SET ora_inizio = '00:00:00';
36         SET ora_fine = '06:00:00';
37     END IF;
38
39     CREATE TEMPORARY TABLE new_tbl
40     SELECT can.nome, prog.titolo, pal.ora_inizio, pal.ora_fine FROM programma prog
41     JOIN programma_preferito prog_pref ON prog.id = prog_pref.id_programma
42     JOIN palinsesto pal ON prog.id = pal.id_programma
43     JOIN canale_preferito can_pref ON pal.id_canale = can_pref.id_canale
44     LEFT JOIN canale can ON pal.id_canale = can.id
45     WHERE pal.ora_inizio >= ora_inizio AND pal.ora_fine <= ora_fine AND can_pref.id_utente = id_param AND prog_pref.id_utente = id_param
```

```

45 WHERE pal.ora_inizio >= ora_inizio AND pal.ora_fine <= ora_fine AND can_pref.id_utente = id_param AND prog_pref.id_utente = id_param
46 ORDER BY pal.ora_inizio ASC;
47
48 SET email_result = "Le ricordiamo che oggi al canale: ";
49 SET count = 0;
50 OPEN curs;
51 iterator: LOOP
52     FETCH curs INTO canale_nome, programma_titolo, ora_inizio, ora_fine;
53     SET count = count + 1;
54     IF exit_loop
55     THEN
56         CLOSE curs;
57         SET exit_loop = FALSE;
58         LEAVE iterator;
59     END IF;
60
61     IF(count > 1)
62     THEN
63         SET email_result = CONCAT(email_result, " invece, ");
64         SET email_result = CONCAT(email_result, " invece, ");
65     END IF;
66     SET email_result = CONCAT(email_result, canale_nome);
67     SET email_result = CONCAT(email_result, " ci sarà in onda il programma ");
68     SET email_result = CONCAT(email_result, programma_titolo);
69     SET email_result = CONCAT(email_result, " a partire dalle ore ");
70     SET email_result = CONCAT(email_result, ora_inizio);
71     SET email_result = CONCAT(email_result, " fino alle ore ");
72     SET email_result = CONCAT(email_result, ora_fine);
73
74     END LOOP iterator;
75
76     DROP TABLE IF EXISTS new_tbl;
77
78 END $
79
80 DELIMITER $

```

```

1
2 • call guida_tv.query_11(1, @email_result);
3 • SELECT @email_result
4

```

Edit Data for @email_result (LONGTEXT)

Binary Text

1 Le ricordiamo che oggi al canale: Rai 1 ci sarà in onda il programma Rambo 1 a partire dalle ore 00:00:00 fino alle ore 02:00:00 invece, Rai 1 ci sarà in onda il programma Rocky a partire dalle ore 02:00:00 fino alle ore 04:00:00 invece, Rai 1 ci sarà in onda il programma Rambo 1 a partire dalle ore 02:00:00 fino alle ore 04:00:00 invece, Rai 1 ci sarà in onda il programma Rocky a partire dalle ore 04:00:00 fino alle ore 06:00:00

12. Procedura_12:

Questa procedura serve per inserire un attore e fare un controllo all'inserimento su eventuali duplicati e anche per controllare la data di nascita, visto che ci serve per identificare un attore

```
6 • -- > [Procedura per inserire un attore.] < --
7 CREATE PROCEDURE `query_12`(IN nome_param VARCHAR(64), IN cognome_param VARCHAR(32), IN data_di_nascita_param DATE)
8 BEGIN
9     IF (data_di_nascita_param > curdate())
10     THEN
11         SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'Forse hai sbagliato la data di nascita';
12     END IF;
13
14     IF EXISTS (SELECT * FROM `guida_tv`.persona p WHERE p.nome = nome_param AND p.cognome = cognome_param AND p.data_di_nascita = data_di_nascita_param)
15     THEN
16         SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'Attore già esistente';
17     END IF;
18
19     INSERT INTO `guida_tv`.persona(`nome`, `cognome`, `data_di_nascita`) VALUES (nome_param, cognome_param, data_di_nascita_param);
20 END $

1 • call guida_tv.inserisci_attore('Sabatino', 'Di Egidio', '1950-10-27');
2
```

15	Elisabetta	Santi	2022-06-01
16	Antonio	Ometto	2022-06-01
17	Gabriele	Curci	2022-06-01
18	Elia	Palange	2022-06-01
19	Fulvio	Zuanni	2022-06-01
20	Sabatino	Di Egidio	1950-10-27
*	NULL	NULL	NULL

13. Procedura_13:

Questa procedura semplicemente serve per inserire un canale

```
4
5 • -- > [Inserimento canale.] < --
6 CREATE PROCEDURE `query_13`(IN nome_param VARCHAR(64), IN numero_param INTEGER)
7 BEGIN
8     INSERT INTO `guida_tv`.canale(`nome`, `numero`) VALUES (nome_param, numero_param);
9 END $
10
```

```
1 • call guida_tv.inserisci_canale('La7', 7);
2
```

	id	nome	numero
▶	1	Rai 1	1
	2	Rai 2	2
	3	Rai 3	3
	4	Canale 4	4
	5	Canale 5	5
	6	Italia 1	6
	7	La7	7
*	NULL	NULL	NULL

14. Procedura_14:

Questa procedura serve per creare un palinsesto facendo dei controlli:

1. sulla data
2. sull'inserimento della data troppo in avanti come scritto nei vincoli sopra
3. sull'inserimento dell'orario di fine del palinsesto facendo un controllo sulla durata sommata all'orario di inizio
4. sull'inserimento di un duplicato

```
5 • -- > [Generazione del palinsesto.] < --
6 CREATE PROCEDURE `query_14`(IN giorno_param DATE, IN ora_inizio_param TIME, IN ora_fine_param TIME, IN id_canale_param INTEGER, IN id_programma_param INTEGER)
7 BEGIN
8
9     DECLARE durata_programma TIME;
10    DECLARE ultimate_ora_fine TIME;
11    SET durata_programma = (SELECT prog.durata FROM `guida_tv`.`programma` prog WHERE prog.id = id_programma_param);
12    SET ultimate_ora_fine = (SELECT p.ora_fine FROM `guida_tv`.`palinsesto` p WHERE p.id_canale = id_canale_param AND p.giorno = giorno_param ORDER BY p.ora_fine DESC LIMIT 1);
13
14    IF (giorno_param > date_add(curdate(), INTERVAL 7 DAY))
15    THEN
16        SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'Non puoi programmare un palinsesto così in avanti';
17    ELSEIF (giorno_param < curdate())
18    THEN
19        SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'Non puoi programmare un palinsesto per giorni precedenti a oggi';
20    ELSEIF (ora_inizio_param + durata_programma > ora_fine_param)
21    THEN
22        SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'Hai sbagliato l\'orario di fine';
23    ELSEIF (ultimate_ora_fine IS NOT NULL AND ultimate_ora_fine > ora_inizio_param)
24    THEN
25        SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'Non puoi inserire un programma nello spazio di un altro';
26    ELSEIF EXISTS
27        (SELECT DISTINCT p.id FROM `guida_tv`.`palinsesto` p WHERE p.giorno = giorno_param AND p.id_canale = id_canale_param
28         AND p.ora_inizio = ora_inizio_param AND p.ora_fine = ora_fine_param)
29    THEN
30        SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'Non puoi inserire un duplicato';
31    END IF;
32
33    INSERT INTO `guida_tv`.`palinsesto` ('giorno', 'ora_inizio', 'ora_fine', 'id_canale', 'id_programma')
34    VALUES (giorno_param, ora_inizio_param, ora_fine_param, id_canale_param, id_programma_param);
35
36 END $
```

```
1 • call guida_tv.genera_palinsesto('2022-07-02', '13:00:00', '14:00:00', 7, 9);
2
```

	id	giorno	ora_inizio	ora_fine	id_canale	id_programma
	2	2022-07-01	02:00:00	04:00:00	1	2
	3	2022-07-01	04:00:00	06:00:00	1	3
	4	2022-07-01	06:00:00	09:16:00	1	4
	5	2022-07-01	09:30:00	11:40:00	1	5
	10	2022-07-01	00:00:00	02:00:00	2	1
	11	2022-07-01	02:00:00	04:00:00	2	2
	12	2022-07-01	04:00:00	06:00:00	2	3
	13	2022-07-01	06:00:00	09:16:00	2	4
	14	2022-07-01	09:30:00	11:40:00	2	8
	15	2022-06-29	00:00:00	02:00:00	1	9
	16	2022-06-29	02:00:00	04:00:00	1	1
	17	2022-06-29	04:00:00	06:00:00	1	2
	19	2022-06-29	11:00:00	13:30:00	1	5
	20	2022-06-29	14:00:00	18:00:00	1	3
	21	2022-07-02	13:00:00	14:00:00	7	9
*	NULL	NULL	NULL	NULL	NULL	NULL

Evento

1. Evento_Autodelete:

Questo è un evento che parte ogni settimana che cancella i programmi, i quali sono stati messi in onda 7 giorni prima, cancello la entry dalla tabella programma e poi tramite le foreign key si cancella in automatico tramite la tabella palinsesto

```
4  DELIMITER $
5
6  •  CREATE
7      EVENT `auto_delete_programs` ON SCHEDULE EVERY 1 WEEK STARTS curdate()
8
9  DO BEGIN
10
11      DECLARE pid INTEGER UNSIGNED;
12      DECLARE curs CURSOR FOR SELECT * FROM new_tbl;
13
14      CREATE TEMPORARY TABLE new_tbl SELECT prog.id
15          FROM `guida_tv`.`programma` prog
16          JOIN `guida_tv`.`palinsesto` pal ON pal.id_programma = prog.id
17          WHERE pal.giorno < DATE_SUB(curdate(), INTERVAL 1 WEEK);
18
19      OPEN curs;
20      LOOP
21          FETCH curs INTO pid;
22          DELETE FROM `guida_tv`.`programma` WHERE (`id` = pid);
23      END LOOP;
24
25      CLOSE curs;
26      DROP TABLE IF EXISTS new_tbl;
27  END $
28  DELIMITER $;
```