

Three degrees of Heath Ledger 4

4.1 Aufgabe

- a. SELECT-Anfrage auf den *database catalog* um alle IMDB-Schema Tabellen auszugeben:

```
1  SELECT TABNAME as Tabellen
2  FROM SYSCAT.TABLES
3  WHERE TABSCHEMA = 'IMDB'
```

- b. SELECT-Anfrage auf den *database catalog* um die Struktur der IMDB.MOVIES-Tabelle auszugeben:

```
1  SELECT COLNAME, TYPENAME, LENGTH
2  FROM SYSCAT.COLUMNS
3  WHERE TABSCHEMA = 'IMDB'
4  AND TABNAME = 'MOVIES' ORDER BY COLNO
```

- c. IMDB.MOVIES-Struktur manuell als **tv_movies_tmp** kopiert:

```
1  CREATE TABLE tv_movies_tmp (
2  TITLE_ID varchar(400),
3  TITLE_TYPE varchar(100),
4  TITLE_TITLE varchar(400),
5  TITLE_YEAR integer,
6  TITLE_NUMERAL integer,
7  TITLE_SUSPENDED decimal,
8  TITLE_ATTRIBUTES varchar(400),
9  EPISODE_SERIES varchar(400),
10 EPISODE_TITLE varchar(400),
11 EPISODE_TITLE_NUMERAL integer,
12 EPISODE_SEASON integer,
13 EPISODE_EPISODE integer,
14 EPISODE_DATE date,
15 YEAR_RANGE varchar(100));
```

4.2 Aufgabe

Struktur einer Tabelle mit

```
1  CREATE TABLE LIKE
```

kopieren: (in diesem Fall die Struktur von **IMDB.ACTORS**)

```
1  CREATE TABLE tv_actors_tmp LIKE IMDB.ACTORS
```

4.3 Aufgabe

- a. Tabellen die überhaupt keine NULL-Werte enthalten:

IMDB.MOVIES:
TITLE_ID
TITLE_TYPE
TITLE_SUSPENDED

IMDB.ACTORS:
NAME_NAME
TITLE_ID
NAME_ID

- b. NOT NULL CONSTRAINT zu **tv_movies_tmp** und zu **tv_actors_tmp** hinzufügen:
tv_movies_tmp

```
1 ALTER TABLE tv_movies_tmp ALTER COLUMN title_id SET NOT NULL
2 ALTER TABLE tv_movies_tmp ALTER COLUMN title_type SET NOT NULL
3 ALTER TABLE tv_movies_tmp ALTER COLUMN title_suspended SET NOT NULL
```

tv_actors_tmp

```
1 ALTER TABLE tv_actors_tmp ALTER COLUMN name_id SET NOT NULL
2 ALTER TABLE tv_actors_tmp ALTER COLUMN name_name SET NOT NULL
3 ALTER TABLE tv_actors_tmp ALTER COLUMN title_id SET NOT NULL
```

4.4 Aufgabe

Struktur von **tv_movies_tmp** und **tv_actors_tmp** kopieren und zwei neue Tabellen **tv_movies** und **tv_actors** mit derselben Struktur erstellen:

```
1 CREATE TABLE tv_movies LIKE tv_movies_tmp
2 CREATE TABLE tv_actors LIKE tv_actors_tmp
```

4.5 Aufgabe

Passende PRIMARY KEYS und FOREIGN KEYS einfügen:

```
1 ALTER TABLE TV_MOVIES ADD PRIMARY KEY (TITLE_ID, TITLE_TYPE);
2 ALTER TABLE TV_ACTORS ADD PRIMARY KEY (NAME_ID, NAME_NAME);
```

4.6 Aufgabe

tv_movies und **tv_actors** mit entsprechenden Daten füllen: Nur *TV movies* aus dem Jahr 2008 und alle ihre *actors* und *actresses* eintragen:

```
1  SELECT * FROM IMDB.MOVIES WHERE TITLE_TYPE = 'TV_movie' AND TITLE_YEAR = '2008'
```

4.7 Aufgabe

materialized view namens *actor_co_occurrences* erstellen, die *tv_movies* und *tv_actors* basiert. Folgende Signatur:

```
1  actor_co_occurrences(actor1, actor2, movie)
```

4.8 Aufgabe

Für *actor1*, *actor2* und *movie* aus der *actor_co_occurrences* einen eigenen Index anlegen.

4.9 Aufgabe

Alle Schauspieler, die eine *Heath Ledger Zahl* von maximal 3 haben. Gib ihre Namen und ihre *Heath Ledger Zahl* aus. Das benutzte SELECT Statement soll **nicht** rekursiv sein!

4.10 Aufgabe

SELECT-Anfrage aus Aufgabe 9 (rekursive Version)