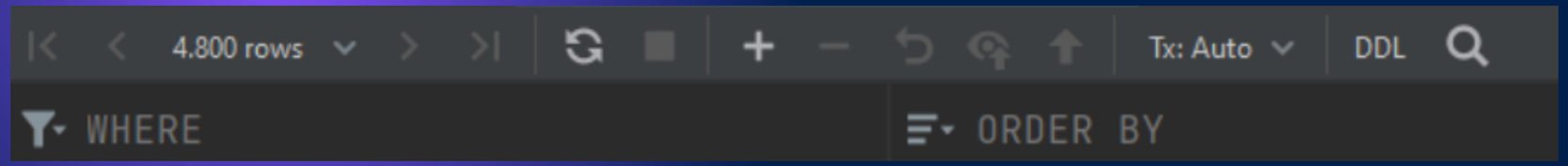
Proyecto Base de Datos

Ronin Carrión

Importación de Archivo



production_companies
[{"name": "Metakwon Filmworks", "id": 65815}]

genres
Action Adventure Fantasy Science Fiction
Adventure Fantasy Action

status
Released
Released
Released
Released
Released
Released
Released
Released
Released
Released

2. Normalizar columnas con campos repetitivos

	status
1	Post Production
2	Released
3	Rumored

	job
1	24 Frame Playback
2	2D Artist
3	2D Supervisor
4	3D Animator
5	3D Artist
6	3D Coordinator
7	3D Modeller
8	3D Supervisor

	department
1	Actors
2	Art
3	Camera
4	Costume & Make-Up
5	Crew
6	Directing
7	Editing
8	Lighting
9	Production

```sql
DROP TABLE IF EXISTS status;
CREATE TABLE status (status VARCHAR(100)) AS SELECT DISTINCT status
FROM movie_dataset.movie_dataset;

ALTER TABLE status
ADD PRIMARY KEY (status);

-- Creación de la relación movie-status
CREATE TABLE status_movie (id INT NOT NULL,
                           status VARCHAR(100))AS SELECT id, st.status AS status
FROM movie_dataset.movie_dataset movie_status, status st
WHERE movie_status.status = st.status;

ALTER TABLE status_movie
ADD FOREIGN KEY (id) REFERENCES movie(id),
    ADD FOREIGN KEY (status) REFERENCES status(status);
```

Eliminación de valores nulos

```sql
DROP TABLE IF EXISTS homepage;
CREATE TABLE homepage(homepage VARCHAR(200)) AS
    SELECT DISTINCT homepage
    FROM movie_dataset.movie_dataset;

DELETE
FROM homepage
WHERE homepage IS NULL;

ALTER TABLE homepage
    ADD PRIMARY KEY (homepage);


CREATE TABLE relacion_homepage (id INT NOT NULL PRIMARY KEY, homepage VARCHAR(200)) AS
    SELECT DISTINCT id, homepage
    FROM movie_dataset.movie_dataset
    WHERE homepage != '' OR homepage IS NOT NULL;
-- Agregamos las claves foráneas de la relacion_homepage a movie y a homepage según corresponda.
ALTER TABLE relacion_homepage
    ADD FOREIGN KEY (id)
        REFERENCES movie(id),
    ADD FOREIGN KEY (homepage)
        REFERENCES homepage(homepage);

```

Columna con valores redundantes

```sql
-- ---------------------------------------- Normalización de Status ----------------------------------------
DROP TABLE IF EXISTS status;
CREATE TABLE status (status VARCHAR(100)) AS SELECT DISTINCT status
FROM movie_dataset.movie_dataset;

ALTER TABLE status
ADD PRIMARY KEY (status);

-- Creación de la relación movie-status
CREATE TABLE status_movie (id INT NOT NULL,
                           status VARCHAR(100))AS SELECT id, st.status AS status
FROM movie_dataset.movie_dataset movie_status, status st
WHERE movie_status.status = st.status;

ALTER TABLE status_movie
ADD FOREIGN KEY (id) REFERENCES movie(id),
    ADD FOREIGN KEY (status) REFERENCES status(status);

```

Columnas con valores no atómicos

```sql
SELECT id, cast, length(cast) as longitud,
length(REPLACE(cast, ' ', '')) as longitudSinEspacios,
length(cast) - length(REPLACE(cast, ' ', '')) as CantidadEspacios,
length(cast) - length(REPLACE(cast, ' ', '')) + 1 as CantidaDePalabras
FROM cast ;
```

	id	cast	longitud	longitudSinEspacios	CantidadEspacios	CantidaDePala... ▲ 1
3891	72766	Edward Burns Kerry Bish\u00ey Marsha Dietlein Caitlin Fitzg…	80	71	9	10
3892	231617	Eric Mabius Kristin Booth Crystal Lowe Geoff Gustafson Benj…	77	68	9	10
3893	126186	Daniel Henney Eliza Coupe Bill Paxton Alan Ruck Zhu Shimao	58	49	9	10
3894	25975	Drew Barrymore Brian Herzlinger Corey Feldman Eric Roberts …	72	63	9	10
3895	49529	Taylor Kitsch Lynn Collins Samantha Morton Willem Dafoe Tho…	75	65	10	11
3896	559	Tobey Maguire Kirsten Dunst James Franco Thomas Haden Churc…	73	63	10	11

```sql

UPDATE cast
SET cast = REPLACE(cast,'Anh Le Guy Pearce', 'Anh Le_Guy Pearce');

```

Columnas Formato Json

```
401  -- ---------------------------------------- Normalización de spoken Languages -----
402
403   # [{"iso_639_1": "en", "name": "English"}]
404
405  DROP PROCEDURE IF EXISTS spoken_languages;
406  DELIMITER $$
407  CREATE PROCEDURE spoken_languages()
408  BEGIN
409   DECLARE done INT DEFAULT FALSE ;
410   DECLARE jsonData json ;
411   DECLARE jsonId varchar(250) ;
412   DECLARE jsonLabel varchar(250) ;
413   DECLARE movieId INT;
414   DECLARE i INT;
415
```

```
416   -- Declarar el cursor
417  DECLARE myCursor
418    CURSOR FOR
419     SELECT id, JSON_EXTRACT(CONVERT(spoken_languages USING UTF8MB4), '$[*]')
420     FROM movie_dataset.movie_dataset;
421   -- Declarar el handler para NOT FOUND
422
423  DECLARE CONTINUE HANDLER
424   FOR NOT FOUND SET done = TRUE ;
425
426   -- Abrir el cursor
427   OPEN myCursor  ;
428  cursorLoop: LOOP
429
430    FETCH myCursor INTO movieId, jsonData;
431      SET i = 0;
432    IF done THEN
433     LEAVE  cursorLoop ;
434    END IF ;
```

Columnas Formato Json

```
436   WHILE(JSON_EXTRACT(jsonData, CONCAT('$[', i, ']')) IS NOT NULL) DO
437     SET jsonId = IFNULL(JSON_EXTRACT(jsonData,  CONCAT('$[', i, '].iso_639_1')), '') ;
438     SET jsonLabel = IFNULL(JSON_EXTRACT(jsonData, CONCAT('$[', i,'].name')), '') ;
439     SET @sql_text = CONCAT('INSERT INTO spoken_languages VALUES (', movieId, ', ', REPLACE(jsonId,'\'',''), ', ', jsonLabel, '); ');
440
441     PREPARE stmt FROM @sql_text;
442     EXECUTE stmt;
443     DEALLOCATE PREPARE stmt;
444     SET i = i + 1;
445   END WHILE;
446
447 END LOOP ;
448
449  CLOSE myCursor ;
450
451 END$$
452 DELIMITER ;
```

```
542 UPDATE movie_dataset.movie_dataset SET crew = CONVERT (
543 REPLACE(REPLACE(REPLACE(REPLACE(REPLACE(REPLACE(REPLACE(crew,
544 '"', '\''),
545 '{\'', '{"'),
546 '\': \'', '": "'),
547 '\', \'', '", "'),
548 '\': ', '": '),
549 ', \'', ', "')
550 USING UTF8mb4 );
```