CODE: Creating Our Tables

Section 13, Lecture 219

-- CREATING THE REVIEWERS TABLE

```
CREATE TABLE reviewers (
    id INT AUTO_INCREMENT PRIMARY KEY,
    first_name VARCHAR(100),
    last_name VARCHAR(100)
);
```

-- CREATING THE SERIES TABLE

```
CREATE TABLE series(
    id INT AUTO_INCREMENT PRIMARY KEY,
    title VARCHAR(100),
    released_year YEAR(4),
    genre VARCHAR(100)
);
```

-- CREATING THE REVIEWS TABLE

```
CREATE TABLE reviews (
    id INT AUTO_INCREMENT PRIMARY KEY,
    rating DECIMAL(2,1),
    series_id INT,
    reviewer_id INT,
    FOREIGN KEY(series_id) REFERENCES series(id),
    FOREIGN KEY(reviewer_id) REFERENCES reviewers(id)
);
```

-- INSERTING A BUNCH OF DATA

```
INSERT INTO series (title, released_year, genre) VALUES
    ('Archer', 2009, 'Animation'),
    ('Arrested Development', 2003, 'Comedy'),
    ("Bob's Burgers", 2011, 'Animation'),
    ('Bojack Horseman', 2014, 'Animation'),
    ("Breaking Bad", 2008, 'Drama'),
    ('Curb Your Enthusiasm', 2000, 'Comedy'),
    ("Fargo", 2014, 'Drama'),
    ('Freaks and Geeks', 1999, 'Comedy'),
    ('General Hospital', 1963, 'Drama'),
    ('Halt and Catch Fire', 2014, 'Drama'),
    ('Malcolm In The Middle', 2000, 'Comedy'),
    ('Pushing Daisies', 2007, 'Comedy'),
    ('Seinfeld', 1989, 'Comedy'),
    ('Stranger Things', 2016, 'Drama');
```

```
INSERT INTO reviewers (first_name, last_name) VALUES
    ('Thomas', 'Stoneman'),
    ('Wyatt', 'Skaggs'),
    ('Kimbra', 'Masters'),
    ('Domingo', 'Cortes'),
    ('Colt', 'Steele'),
    ('Pinkie', 'Petit'),
    ('Marlon', 'Crafford');


INSERT INTO reviews(series_id, reviewer_id, rating) VALUES
    (1,1,8.0),(1,2,7.5),(1,3,8.5),(1,4,7.7),(1,5,8.9),
    (2,1,8.1),(2,4,6.0),(2,3,8.0),(2,6,8.4),(2,5,9.9),
    (3,1,7.0),(3,6,7.5),(3,4,8.0),(3,3,7.1),(3,5,8.0),
    (4,1,7.5),(4,3,7.8),(4,4,8.3),(4,2,7.6),(4,5,8.5),
    (5,1,9.5),(5,3,9.0),(5,4,9.1),(5,2,9.3),(5,5,9.9),
    (6,2,6.5),(6,3,7.8),(6,4,8.8),(6,2,8.4),(6,5,9.1),
    (7,2,9.1),(7,5,9.7),
    (8,4,8.5),(8,2,7.8),(8,6,8.8),(8,5,9.3),
    (9,2,5.5),(9,3,6.8),(9,4,5.8),(9,6,4.3),(9,5,4.5),
    (10,5,9.9),
    (13,3,8.0),(13,4,7.2),
    (14,2,8.5),(14,3,8.9),(14,4,8.9);
```

CODE: TV Joins Challenge 1 Solution

Section 13, Lecture 221

-- TV Joins Challenge 1 SOLUTION

```
SELECT
    title,
    rating
FROM series
JOIN reviews
    ON series.id = reviews.series_id;
```

CODE: TV Joins Challenge 2 SOLUTION

Section 13, Lecture 223

-- Challenge 2 AVG rating

```
SELECT
    title,
    AVG(rating) as avg_rating
FROM series
JOIN reviews
    ON series.id = reviews.series_id
GROUP BY series.id
ORDER BY avg_rating;
```

CODE: TV Joins Challenge 3 SOLUTION

Section 13, Lecture 225

-- CHALLENGE 3 - Two Solutions

```
SELECT
    first_name,
    last_name,
    rating
FROM reviewers
INNER JOIN reviews
    ON reviewers.id = reviews.reviewer_id;
```

```
SELECT
    first_name,
    last_name,
    rating
FROM reviews
INNER JOIN reviewers
    ON reviewers.id = reviews.reviewer_id;
```

CODE: TV Joins Challenge 4 SOLUTION

Section 13, Lecture 227

-- CHALLENGE 4 - UNREVIEWED SERIES

```
SELECT title AS unreviewed_series
FROM series
LEFT JOIN reviews
    ON series.id = reviews.series_id
WHERE rating IS NULL;
```

CODE: TV Joins Challenge 5 SOLUTION

Section 13, Lecture 229

-- Challenge 5 - GENRE AVG RATINGS

```
SELECT genre,
       Round(Avg(rating), 2) AS avg_rating
FROM   series
       INNER JOIN reviews
               ON series.id = reviews.series_id
GROUP  BY genre;
```

CODE: TV Joins Challenge 6 SOLUTION

Section 13, Lecture 231

-- CHALLENGE 6 - Reviewer Stats

```
SELECT first_name,
       last_name,
       Count(rating)                               AS COUNT,
       Ifnull(Min(rating), 0)                      AS MIN,
       Ifnull(Max(rating), 0)                      AS MAX,
       Round(Ifnull(Avg(rating), 0), 2)            AS AVG,
       IF(Count(rating) > 0, 'ACTIVE', 'INACTIVE') AS STATUS
FROM   reviewers
       LEFT JOIN reviews
              ON reviewers.id = reviews.reviewer_id
GROUP  BY reviewers.id;
```

-- CHALLENGE 6 - Reviewer Stats With POWER USERS

```
SELECT first_name,
       last_name,
       Count(rating)                    AS COUNT,
       Ifnull(Min(rating), 0)           AS MIN,
       Ifnull(Max(rating), 0)           AS MAX,
       Round(Ifnull(Avg(rating), 0), 2) AS AVG,
       CASE
         WHEN Count(rating) >= 10 THEN 'POWER USER'
         WHEN Count(rating) > 0 THEN 'ACTIVE'
         ELSE 'INACTIVE'
       end                              AS STATUS
FROM   reviewers
       LEFT JOIN reviews
              ON reviewers.id = reviews.reviewer_id
GROUP  BY reviewers.id;
```

CODE: TV Joins Challenge 7 SOLUTION

Section 13, Lecture 233

-- CHALLENGE 7 - 3 TABLES!

```
SELECT
    title,
    rating,
    CONCAT(first_name,' ', last_name) AS reviewer
FROM reviewers
INNER JOIN reviews
    ON reviewers.id = reviews.reviewer_id
INNER JOIN series
    ON series.id = reviews.series_id
ORDER BY title;
```