odspro Subs Workflow Database

Project Demonstration

Oliver D Siu

July 16, 2025

Abstract

This project demonstrates relational database design and SQL expertise through a donghua fansubbing workflow management system. Built from scratch with eight normalized tables and optimized queries, the database tracks 80+ episodes revealing significant production irregularities: 8-24 episodes per season, runtime variations (14-44 minutes), and fluctuating technical specifications. Advanced SQL analysis quantifies donghua's unique workflow challenges compared to standardized anime production, showcasing database implementation skills including proper indexing, foreign key constraints, and complex query optimization for real-world data management.

Project Overview

Specifications

Database:

- Name: `subbing.db`
- Type: SQLite relational database
- Tables: 8 normalized tables with foreign key constraints

Data Source:

This database incorporates data extracted from my fan subtitling (fansub) workflow, where I am recognized in the community as "odspro". The dataset encompasses 80+ episodes across multiple series and seasons.

Implementation Scripts:

- `fansub_export.R` - Data extraction and processing
- `fansub_db_init.R` - Database initialization and table creation

Dependencies:

- R libraries: `knitr`, `tidyverse`, `fs`, `DBI`, `RSQLite`, `RMariaDB`
- External tools: `ffprobe` (from FFmpeg toolkit) for video metadata extraction

Project Background

Recent developments in the media translation industry demonstrate significant improvements in the quality, quantity, and delivery speed of official English translations for foreign content. This trend is particularly pronounced within Japanese animated media (anime), where the proliferation of official translation services has resulted in a substantial decline in fan subtitling (fansub) communities due to several key factors:

1. Professional translation opportunities have attracted experienced fansubbers, who leverage their community expertise for commercial roles.
2. The accessibility and affordability of official translations provide consumers with convenient legal alternatives to fan-produced content.
3. Expectations of quality have risen, requiring fansubbers to demonstrate clear superiority over official translations to maintain relevance.

This third factor manifests across multiple aspects. Beyond enhanced translation quality, fan subtitles must incorporate advanced technical features including song lyric translations, sign translations, and karaoke animations. These elevated standards are sustainable within the anime community where official subtitles can supplement fansubbing workflows, but the Chinese animation (donghua) ecosystem lacks comparable infrastructure.

The donghua translation landscape presents distinct challenges. A majority of donghua remains without official English translations, and existing translations frequently exhibit quality issues, often relying on machine translation with minimal human oversight. Some platforms explicitly acknowledge their use of automated translation systems without review processes. This translation gap has generated renewed demand for fansubbers, yet quality expectations from the anime community have transferred to donghua despite the absence of supporting infrastructure. These elevated standards create significant barriers for newcomers, as sustainable fansubbing typically requires collaborative teams to distribute specialized tasks such as video encoding, multiplexing, timing, typesetting, styling, subtitle animation, and related technical processes. Consequently, solo fansubbers must simultaneously manage translation responsibilities alongside comprehensive technical production workflows.

This project demonstrates the multifaceted workflow management challenges inherent in donghua fansubbing operations. A relational database was selected based on the structured characteristics of the production workflow. Experience from anime fansubbing workflows provides a foundation for consistent operational patterns which Chinese animation productions also follow. This standardization creates data consistency that is well-suited for relational database implementation.

Schema

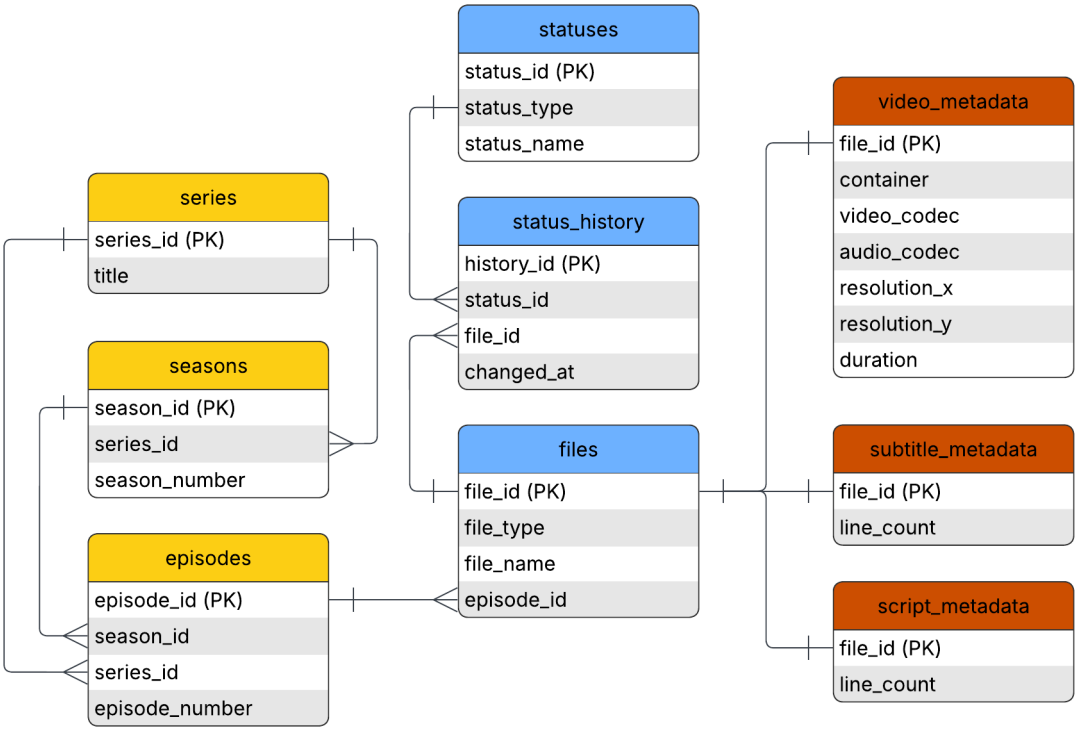


Figure 1: Database Schema Diagram

Note:
- (PK) indicates Primary Key
- (FK) indicates Foreign Key

Tables:

series
- `series_id` (PK)
- `title`

seasons
- `season_id` (PK)
- `series_id` (FK to `series.series_id`)
- `season_number`

episodes
- `episode_id` (PK)
- `season_id` (FK to `seasons.season_id`)
- `series_id` (FK to `series.series_id`)
- `episode_number`

statuses
- `status_id` (PK)
- `status_type`
- `status_name`

status_history
- `history_id` (PK)
- `file_id` (FK to `files.file_id`)
- `status_id` (FK to `statuses.status_id`)
- `changed_at`

files
- `file_id` (PK)
- `file_type`
- `file_name`
- `episode_id` (FK to `episodes.episode_id`)

video_metadata
- `file_id` (PK)
- `container`
- `video_codec`
- `audio_codec`
- `resolution_x`
- `resolution_y`
- `duration`

subtitle_metadata
- `file_id` (PK)
- `line_count`

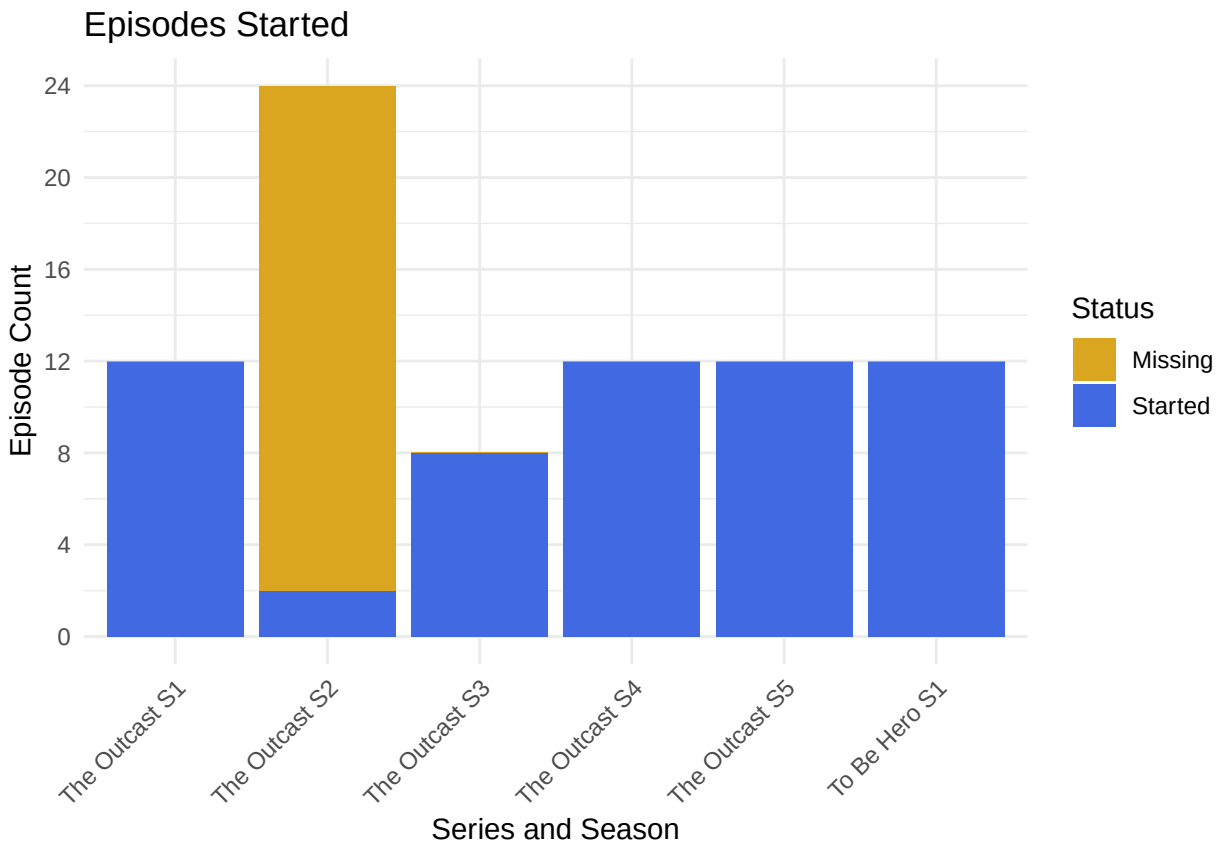
script_metadata
- `file_id` (PK)
- `line_count`

Relationships:
- One series has many seasons
- One season has many episodes
- One series has many episodes
- One episode has many files
- One status can have many `status_history` entries
- One file can have many `status_history` entries
- Each file can have only one `video_metadata` or `subtitle_metadata` or `script_metadata` entry

Sample Queries

Query 1 – Which episodes have not been subtitled?

```sql
-- Query to count total episodes and missing subtitles by series and season
-- Uses LEFT JOIN to identify episodes without subtitle files
SELECT
    s.title AS series,
    se.season_number,
    COUNT(e.episode_id) AS total_episodes,
    COUNT(
        CASE
            WHEN f.file_id IS NULL THEN 1
        END
    ) AS missing_subtitles
FROM episodes AS e
JOIN seasons AS se
    ON e.season_id = se.season_id
JOIN series AS s
    ON se.series_id = s.series_id
LEFT JOIN files AS f
    ON f.episode_id = e.episode_id AND f.file_type = 'subtitle'
GROUP BY s.title, se.season_number
ORDER BY s.title, se.season_number;
```



The bar chart visualizes the distribution of episodes with and without subtitle file initialization across series and seasons. The analysis reveals that The Outcast Season 2 exhibits substantial missing subtitle

initialization. This large amount of missing data may impact subsequent query calculations. Episode counts per season demonstrate considerable variability. A typical season contains 12 episodes, while Season 2 contains 24 episodes and Season 3 contains 8 episodes. These variations indicate inconsistent production scheduling patterns.

Query 2 – Which are the 10 most recent status updates for subtitle files?

```sql
-- Query to get the latest status updates for subtitle files
-- Uses CTE with ROW_NUMBER() to find most recent status per file
WITH latest_status AS (
    SELECT
        f.file_id,
        st.status_type,
        st.status_name,
        sh.changed_at,
        f.file_type,
        ROW_NUMBER() OVER (
            PARTITION BY f.file_id
            ORDER BY sh.changed_at DESC
        ) AS rn
    FROM files AS f
    JOIN status_history AS sh
        ON f.file_id = sh.file_id
    JOIN statuses AS st
        ON sh.status_id = st.status_id
    WHERE f.file_type = 'subtitle'
)
SELECT
    s.title AS series,
    se.season_number,
    e.episode_number,
    ls.file_type AS table_source,
    ls.status_type,
    ls.status_name,
    ls.changed_at
FROM latest_status AS ls
JOIN episodes AS e
    ON ls.file_id = e.episode_id
JOIN seasons AS se
    ON e.season_id = se.season_id
JOIN series AS s
    ON se.series_id = s.series_id
WHERE ls.rn = 1
ORDER BY s.title, se.season_number, e.episode_number
LIMIT 10;
```

Table 1: 10 Most Recent Status Updates

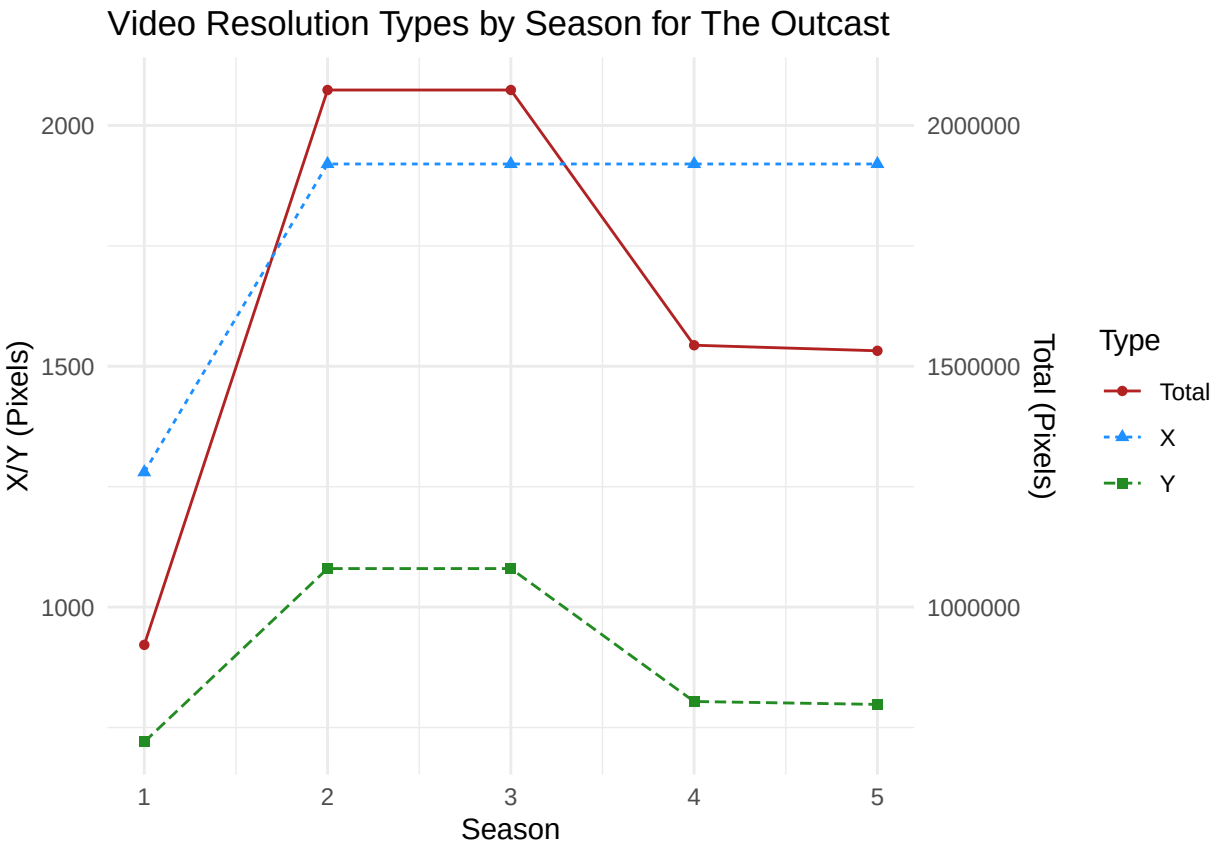
Series	Season	Episode	File Type	Status Type	Status	Changed At
The Outcast	1	1	subtitle	revision	in progress	2025-02-23 03:23:00
The Outcast	1	2	subtitle	revision	in progress	2025-02-23 03:23:00

Series	Season	Episode	File Type	Status Type	Status	Changed At
The Outcast	1	3	subtitle	revision	in progress	2025-02-23 03:23:00
The Outcast	1	4	subtitle	revision	in progress	2025-02-23 03:23:00
The Outcast	1	5	subtitle	revision	in progress	2025-02-23 03:23:00
The Outcast	1	6	subtitle	revision	in progress	2025-02-23 03:23:00
The Outcast	1	7	subtitle	revision	in progress	2025-02-23 03:23:00
The Outcast	1	8	subtitle	revision	in progress	2025-02-23 03:23:00
The Outcast	1	9	subtitle	revision	in progress	2025-02-23 03:23:00
The Outcast	1	10	subtitle	revision	in progress	2025-02-23 03:23:00

This query returns commonly requested information regarding subtitle file status tracking. SQL backends are frequently implemented to support web applications by offloading computational tasks from client devices. The development of this query was motivated by documented incidents where fansubbing organizations received criticism for lacking such informational capabilities on their platforms.

Query 3 – Has video resolution increased with each season of The Outcast?

```
-- Query to find the highest video resolution for each season of 'The Outcast'
-- Uses CTE with ROW_NUMBER() to rank resolutions by total pixel count
WITH season_resolutions AS (
    SELECT
        s.title AS series,
        se.season_number,
        vm.resolution_x,
        vm.resolution_y,
        (vm.resolution_x * vm.resolution_y) AS resolution_total,
        ROW_NUMBER() OVER (
            PARTITION BY se.season_number
            ORDER BY (vm.resolution_x * vm.resolution_y) DESC
        ) AS rn
    FROM series AS s
    JOIN seasons AS se
        ON s.series_id = se.series_id
    JOIN episodes AS e
        ON se.season_id = e.season_id
    JOIN files AS f
        ON e.episode_id = f.episode_id AND f.file_type = 'video'
    JOIN video_metadata AS vm
        ON f.file_id = vm.file_id
    WHERE s.title = 'The Outcast'
)
SELECT
    series,
    season_number,
    resolution_x,
    resolution_y,
    resolution_total
FROM season_resolutions
WHERE rn = 1
ORDER BY season_number;
```

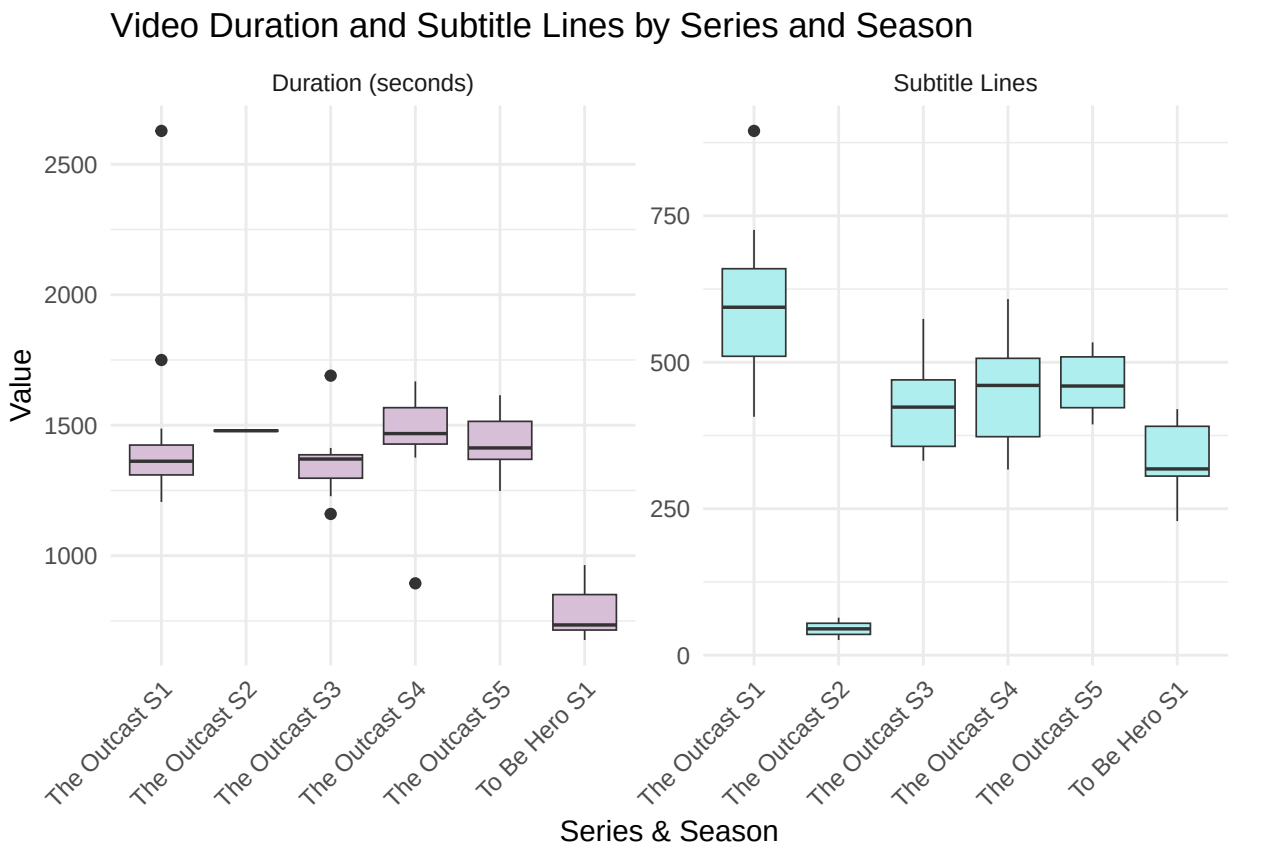


The visualization displays video resolution metrics across seasons for The Outcast series. The analysis examines peak resolution values for each season due to internal resolution variations from inconsistent production quality. Video resolution has not demonstrated consistent improvement across seasons. Maximum pixel resolution of 1920x1080 was achieved in Seasons 2 and 3. Subsequent seasons maintained 1920 pixel horizontal resolution while reducing vertical resolution for artistic ultrawide aspect ratios, which compromised overall pixel count.

Query 4 – Is there a correlation between subtitle line counts and video durations?

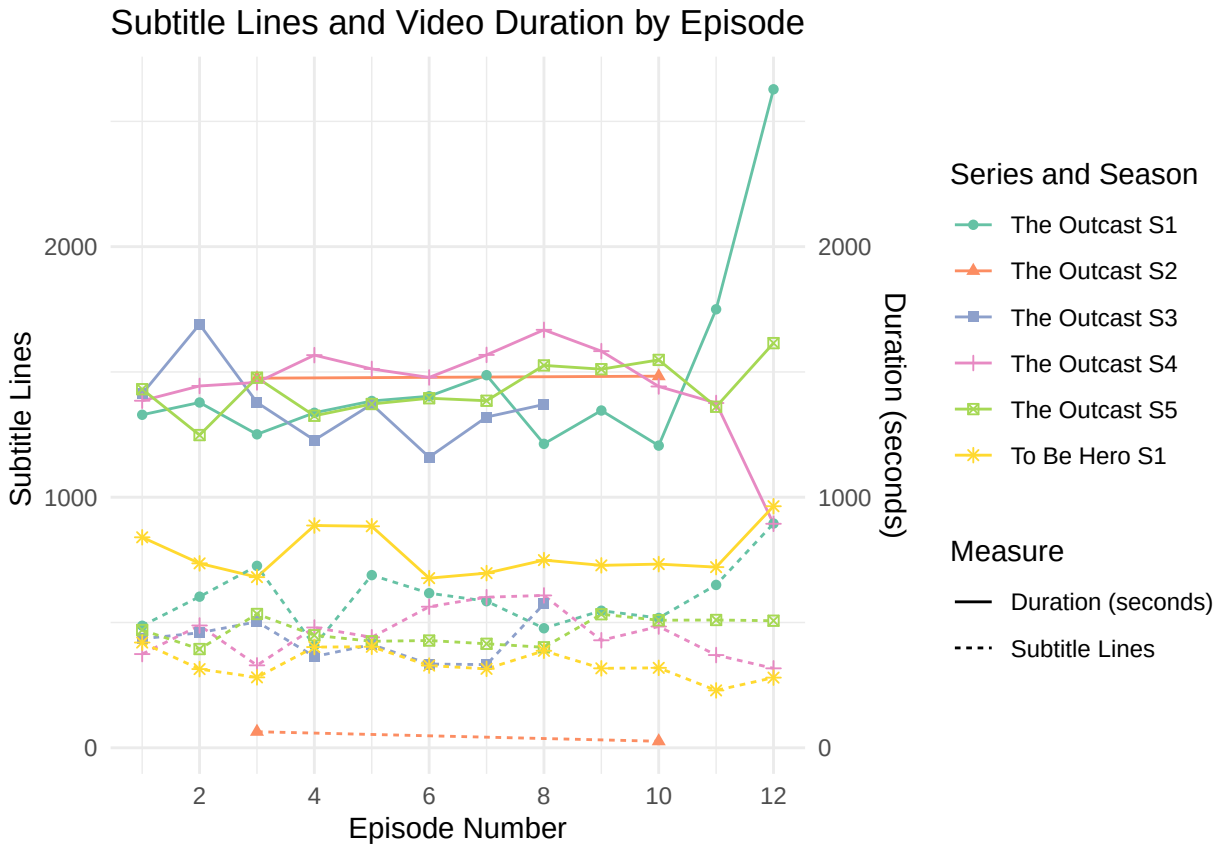
```
-- Query to compare subtitle line counts with video durations
-- Uses two CTEs to join subtitle and video metadata, converts duration to seconds
WITH
    subtitle AS (
        SELECT
            e.episode_id,
            sm.line_count
        FROM episodes AS e
        JOIN files AS f
            ON e.episode_id = f.episode_id AND f.file_type = 'subtitle'
        JOIN subtitle_metadata AS sm
            ON f.file_id = sm.file_id
    ),
    video AS (
        SELECT
            e.episode_id,
            vm.duration
```

```
        FROM episodes AS e
        JOIN files f
            ON e.episode_id = f.episode_id AND f.file_type = 'video'
        JOIN video_metadata vm
            ON f.file_id = vm.file_id
    )
SELECT
    s.title AS series,
    se.season_number,
    e.episode_number,
    subtitle.line_count AS subtitle_lines,
    (CAST(SUBSTR(video.duration, 1, 2) AS INTEGER) * 3600 +
     CAST(SUBSTR(video.duration, 4, 2) AS INTEGER) * 60 +
     CAST(SUBSTR(video.duration, 7, 2) AS INTEGER)) AS video_duration_seconds
FROM series AS s
JOIN seasons AS se
    ON s.series_id = se.series_id
JOIN episodes AS e
    ON se.season_id = e.season_id
JOIN subtitle
    ON e.episode_id = subtitle.episode_id
LEFT JOIN video
    ON e.episode_id = video.episode_id
ORDER BY s.title, se.season_number, e.episode_number;
```



The boxplot analysis examines the distribution of subtitle line counts and video duration across series and seasons. While The Outcast Season 1 typically exhibits the highest subtitle line counts, its video duration

remains comparatively shorter than other seasons. This discrepancy may result from production changes in later seasons, which incorporated baked advertisements into video files, thereby extending runtime. To Be Hero demonstrates significantly shorter runtime compared to The Outcast, yet maintains comparable subtitle line density. This pattern may reflect differences in genre and show styles.

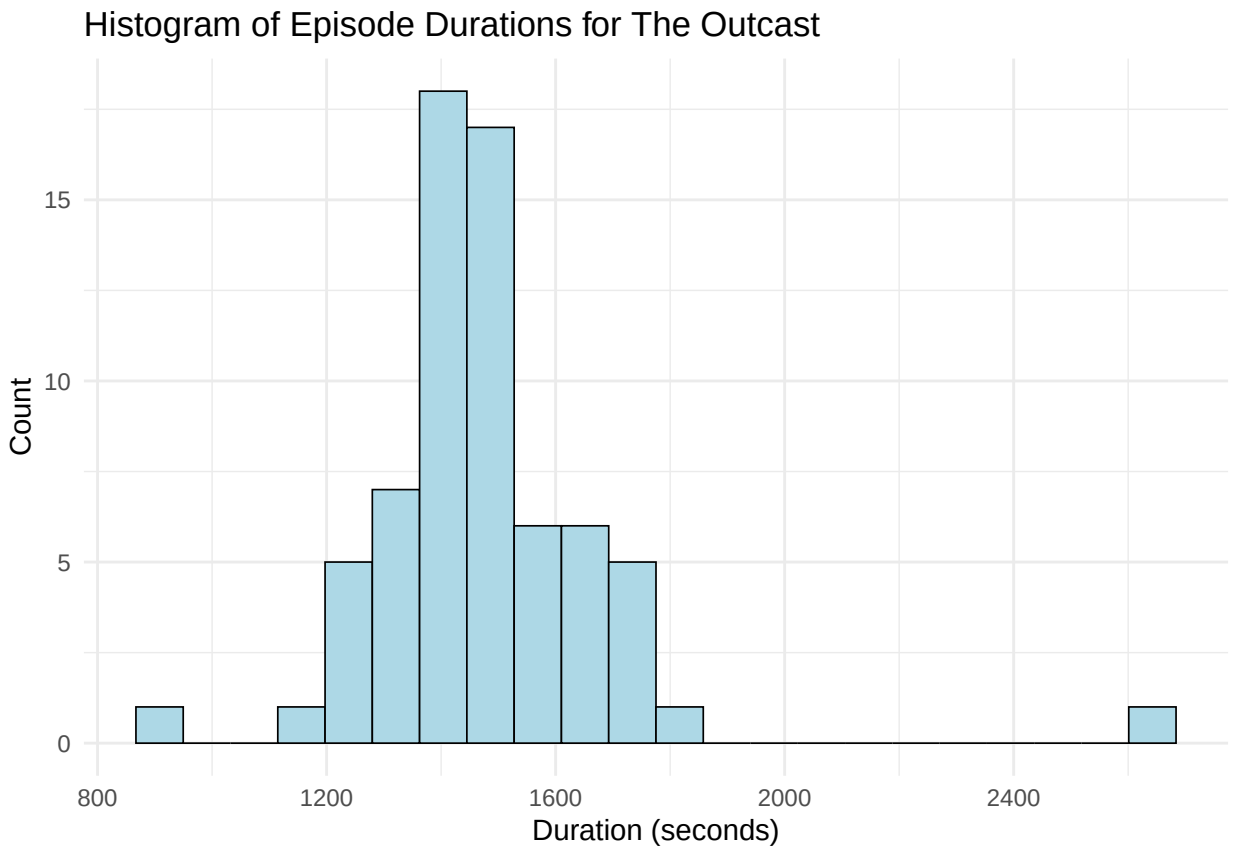


The analysis reveals distinct patterns in runtime and subtitle density throughout each season's progression. The Outcast Seasons 1 and 5 demonstrate extended runtime for season finales, while Season 4 concludes with the shortest episode duration in the entire series. To Be Hero Season 1 similarly implements extended runtime for its season finale, suggesting a common production strategy across different series.

Query 5 – How long are the episodes of The Outcast? Are there any outliers?

```sql
-- Query to analyze episode durations for 'The Outcast' series
-- Converts HH:MM:SS duration format to seconds for easier analysis
SELECT
    se.season_number,
    e.episode_number,
    vm.duration,
    (CAST(SUBSTR(vm.duration, 1, 2) AS INTEGER) * 3600 +
     CAST(SUBSTR(vm.duration, 4, 2) AS INTEGER) * 60 +
     CAST(SUBSTR(vm.duration, 7, 2) AS FLOAT)) AS duration_seconds
FROM series AS s
JOIN seasons AS se
    ON s.series_id = se.series_id
JOIN episodes AS e
    ON se.season_id = e.season_id
```

```sql
JOIN files AS f
    ON e.episode_id = f.episode_id AND f.file_type = 'video'
JOIN video_metadata AS vm
    ON f.file_id = vm.file_id
WHERE s.title = 'The Outcast'
ORDER BY duration_seconds DESC;
```



The analysis of The Outcast episode durations reveals a concentration around 1400-1500 seconds (approximately 24 minutes). The distribution exhibits slight right-skewness with two notable outliers identified in the analysis.

Table 2: Top 5 Longest Episodes

Season	Episode	Duration (HH:MM:SS)	Duration (seconds)
1	12	00:43:48	2628
2	5	00:29:56	1796
2	12	00:29:30	1770
1	11	00:29:10	1750
2	4	00:28:54	1734

Table 3: Top 5 Shortest Episodes

	Season	Episode	Duration (HH:MM:SS)	Duration (seconds)
64	3	4	00:20:28	1228

	Season	Episode	Duration (HH:MM:SS)	Duration (seconds)
65	1	8	00:20:13	1213
66	1	10	00:20:06	1206
67	3	6	00:19:20	1160
68	4	12	00:14:54	894

The outliers are identified as Season 1 Episode 12 with an extended 43:48 runtime and Season 4 Episode 12 with a condensed 14:54 runtime. Season 4 Episode 12 represents the shortest episode in the series despite containing embedded advertisements.

Query 6 – When did episodes receive their first revision? How long did it take?

```sql
-- Query to calculate time between episode completion and first revision
-- Uses multiple CTEs and window functions to track status changes over time
WITH episode_status_times AS (
    SELECT
        f.episode_id,
        e.episode_number,
        se.season_number,
        s.title AS series,
        sh.changed_at,
        st.status_type,
        ROW_NUMBER() OVER (
            PARTITION BY f.episode_id, st.status_type
            ORDER BY sh.changed_at DESC
        ) AS rn_all,
        ROW_NUMBER() OVER (
            PARTITION BY f.episode_id, st.status_type
            ORDER BY sh.changed_at
        ) AS rn_revision
    FROM status_history AS sh
    JOIN statuses AS st
        ON sh.status_id = st.status_id
    JOIN files AS f
        ON sh.file_id = f.file_id
    JOIN episodes AS e
        ON f.episode_id = e.episode_id
    JOIN seasons AS se
        ON e.season_id = se.season_id
    JOIN series AS s
        ON se.series_id = s.series_id
    WHERE st.status_type IN ('all', 'revision')
)
SELECT
    series,
    season_number,
    episode_number,
    MAX(
        CASE
            WHEN status_type = 'all'
            AND rn_all = 1
```

```
            THEN changed_at
        END
    ) AS all_complete_time,
    MIN(
        CASE
            WHEN status_type = 'revision'
            AND rn_revision = 1
            THEN changed_at
        END
    ) AS revision_time,
    ROUND(
        JULIANDAY(
            MIN(
                CASE
                    WHEN status_type = 'revision'
                    AND rn_revision = 1
                    THEN changed_at
                END
            )
        ) -
        JULIANDAY(
            MAX(
                CASE
                    WHEN status_type = 'all'
                    AND rn_all = 1
                    THEN changed_at
                END
            )
        ),
        2
    ) AS days_between
FROM episode_status_times
GROUP BY series, season_number, episode_number
HAVING all_complete_time IS NOT NULL AND revision_time IS NOT NULL
ORDER BY days_between DESC
LIMIT 10;
```

Table 4: 10 Longest Time Between Episode Completion and Revision

Series	Season	Episode	Completion Time	Revision Time	Days Between
The Outcast	1	1	2021-01-20 09:47:00	2025-02-23 03:23:00	1494.73
The Outcast	1	2	2021-01-20 09:47:00	2025-02-23 03:23:00	1494.73
The Outcast	1	3	2021-01-20 09:47:00	2025-02-23 03:23:00	1494.73
The Outcast	1	4	2021-01-20 09:47:00	2025-02-23 03:23:00	1494.73
The Outcast	1	5	2021-01-20 09:47:00	2025-02-23 03:23:00	1494.73
The Outcast	1	6	2021-01-20 09:47:00	2025-02-23 03:23:00	1494.73
The Outcast	1	7	2021-01-20 09:47:00	2025-02-23 03:23:00	1494.73
The Outcast	1	8	2021-01-20 09:47:00	2025-02-23 03:23:00	1494.73
The Outcast	1	9	2021-01-20 09:47:00	2025-02-23 03:23:00	1494.73
The Outcast	1	10	2021-01-20 09:47:00	2025-02-23 03:23:00	1494.73

The analysis examines the time between fansub completion and first revision for all episodes that received

revision treatment. Season 1 experienced an interval of 1494.73 days (approximately 4 years) between initial fansub release and first revision. The data indicates that only 24 of the 80 episodes in the dataset have undergone revision processes, demonstrating that the majority of fansub content has not received post-release refinement.

Findings

Production Variability and Workflow Impact

This database analysis reveals significant production inconsistencies in donghua that create unique workflow challenges for fansubbers. The data demonstrates extreme variability: 8-24 episodes per season, runtime fluctuations from 14:54 to 43:48 minutes, and non-standardized video resolutions requiring constant subtitle script adjustments. These irregularities force fansubbers to continuously adapt their workflows, unlike the standardized anime production patterns.

These inconsistencies have measurable impacts on translator workload: subtitle line counts can double or halve between episodes, and resolution changes demand technical reconfiguration for each release. The resource-intensive nature of donghua fansubbing is further evidenced by status tracking data showing that episodes often take years to receive revisions. Managing such complex and unpredictable workflows requires systematic data organization and analysis.

Database Solution

Despite the complexity, relational database management proves valuable for tracking these multifaceted workflows. The normalized schema successfully handles irregular production patterns while maintaining data integrity through proper foreign key constraints. Advanced SQL queries reveal production insights that would be difficult to extract manually. While a relational database might seem excessive for fansubbing projects, the workflow irregularities documented here justify this approach, especially given that machine learning tools are already prevalent in the donghua fansubbing community.

Community Implications

The data underscores why donghua fansubbing faces translator shortages compared to anime. Without official English support and given China's large domestic market, the burden falls entirely on volunteer translators managing technically demanding, unpaid work. Database tools could help streamline these workflows, potentially reducing barriers to entry for new fansubbers.

Given these challenges, consumers need greater patience with fansubbers. The donghua community needs volunteers for video encoding, multiplexing, timing, typesetting, and styling—technical skills that directly address workflow bottlenecks identified in this analysis. Collaborative support and database-driven workflow optimization may help the donghua fansubbing community thrive and bring more quality translations to international audiences.