In-Class Task: SQL Indexes - Streaming Service Platform

Scenario

You work for a video streaming platform similar to Netflix. The database contains millions of viewing records, and users are complaining about slow load times when browsing content and viewing their watch history. You need to optimize the database with appropriate indexes.

Database Setup (Provided)

sql

```
CREATE TABLE users (
    user_id INT PRIMARY KEY,
    username VARCHAR(50),
    email VARCHAR(100),
    subscription_type VARCHAR(20),
    country VARCHAR(50)
);

CREATE TABLE videos (
    video_id INT PRIMARY KEY,
    title VARCHAR(200),
    genre VARCHAR(50),
    release_year INT,
    duration_minutes INT,
    rating DECIMAL(3,1)
);

CREATE TABLE watch_history (
    watch_id INT PRIMARY KEY,
    user_id INT,
    video_id INT,
    watch_date TIMESTAMP,
    watch_duration_minutes INT,
    completed BOOLEAN,
    FOREIGN KEY (user_id) REFERENCES users(user_id),
    FOREIGN KEY (video_id) REFERENCES videos(video_id)
);

-- Sample data already inserted (1M+ users, 50K+ videos, 100M+ watch records)
```

Tasks (Complete within 10 minutes)

Task 1: Optimize User Watch History (3 points)

Users frequently view "My Watch History" which queries their recent viewing activity. Create an index to optimize this common query:

sql
```
SELECT v.title, w.watch_date, w.completed
FROM watch_history w
JOIN videos v ON w.video_id = v.video_id
WHERE w.user_id = 12345
ORDER BY w.watch_date DESC;
```
Your Index:

Question: Should this be a single-column or multicolumn index? _____

Task 2: Expression Index for Search (3 points)

The search feature allows users to find videos by title, ignoring case and leading/trailing spaces. Create an expression index:

sql
```
-- Write your SQL here:
```
Question: Write a SELECT query that would utilize this index:

Task 3: Analyze Existing Indexes (2 points)

The following indexes currently exist on `watch_history`:

sql
```
idx1: CREATE INDEX wh_user_idx ON watch_history(user_id);
idx2: CREATE INDEX wh_user_date_idx ON watch_history(user_id, watch_date);
idx3: CREATE INDEX wh_date_idx ON watch_history(watch_date);
```
Question A: Which index is redundant and should be dropped? _____

Question B: Write the DROP statement:

Task 4: Conditional Index (2 points)

The recommendation engine only analyzes completed views (completed = TRUE), which represents about 40% of all watch records. Create a partial index:

sql
```
-- Write your SQL here:
```

Question: What is the main advantage of using a partial index here?