Project Report: Job Data Analysis

1. Project Description

The goal of this project is to analyze job review data, focusing on metrics such as jobs reviewed over time, throughput, language share, and detection of duplicate records. This analysis aims to uncover patterns in job processing, improve organizational efficiency, and provide data-driven insights for decision-making. To handle the analysis, I structured and imported the dataset into a MySQL database, enabling effective querying and data manipulation for each analytical task.

2. Approach

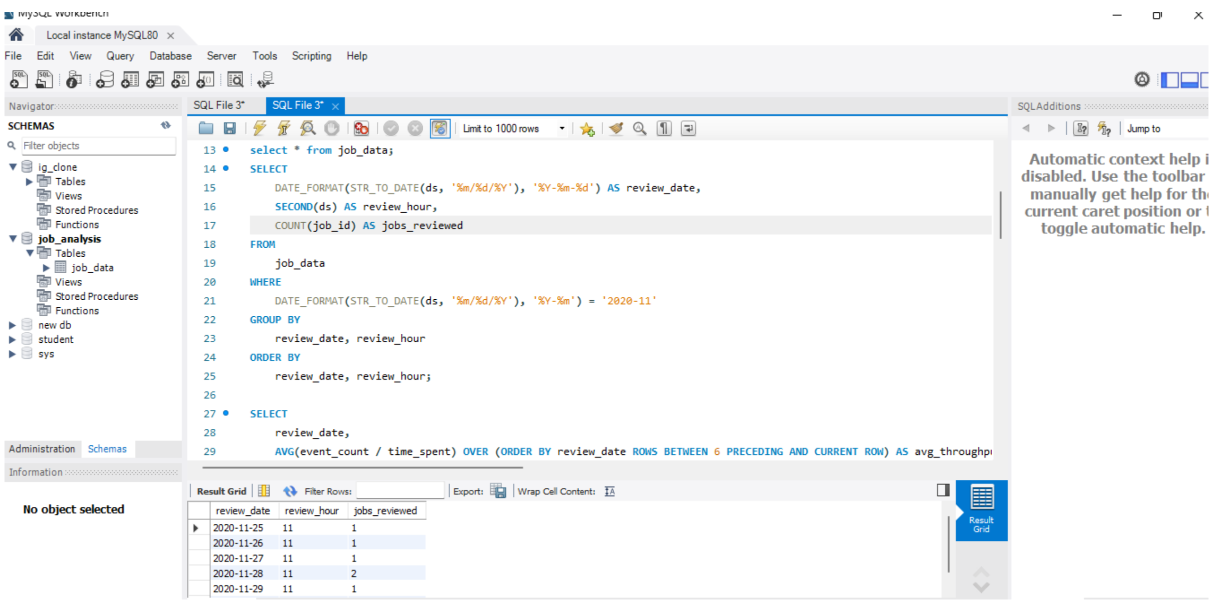
The approach involved several key steps:

- **Data Preparation**: The CSV file was imported into MySQL Workbench, creating a structured job_data table to facilitate SQL queries.
- **Query Execution**: For each case study question, I wrote SQL queries to retrieve the required metrics. Each query focused on extracting and summarizing data from the job_data table to meet specific objectives.
- **Analysis**: I calculated various metrics, including jobs reviewed per hour, a 7-day rolling average for throughput, language share percentages, and identified duplicate records. This approach enabled thorough examination and provided a clear view of the dataset's behavior.

3. Tech-Stack Used

- **MySQL Workbench**: Used as the primary tool for database management, data import, and SQL query execution.

Task 1: Jobs Reviewed Over Time



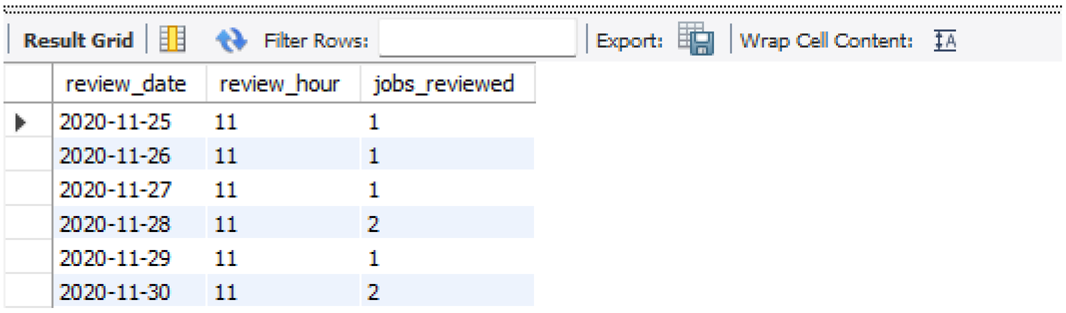
```sql
select * from job_data;
SELECT
    DATE_FORMAT(STR_TO_DATE(ds, '%m/%d/%Y'), '%Y-%m-%d') AS review_date,
    SECOND(ds) AS review_hour,
    COUNT(job_id) AS jobs_reviewed
FROM
    job_data
WHERE
    DATE_FORMAT(STR_TO_DATE(ds, '%m/%d/%Y'), '%Y-%m') = '2020-11'
GROUP BY
    review_date, review_hour
ORDER BY
    review_date, review_hour;

SELECT
    review_date,
    AVG(event_count / time_spent) OVER (ORDER BY review_date ROWS BETWEEN 6 PRECEDING AND CURRENT ROW) AS avg_throughp
```

review_date	review_hour	jobs_reviewed
2020-11-25	11	1
2020-11-26	11	1
2020-11-27	11	1
2020-11-28	11	2
2020-11-29	11	1

```sql
select * from job_data;
SELECT
    DATE_FORMAT(STR_TO_DATE(ds, '%m/%d/%Y'), '%Y-%m-%d') AS review_date,
    SECOND(ds) AS review_hour,
    COUNT(job_id) AS jobs_reviewed
FROM
    job_data
WHERE
    DATE_FORMAT(STR_TO_DATE(ds, '%m/%d/%Y'), '%Y-%m') = '2020-11'
GROUP BY
    review_date, review_hour
ORDER BY
    review_date, review_hour;
```

Insights

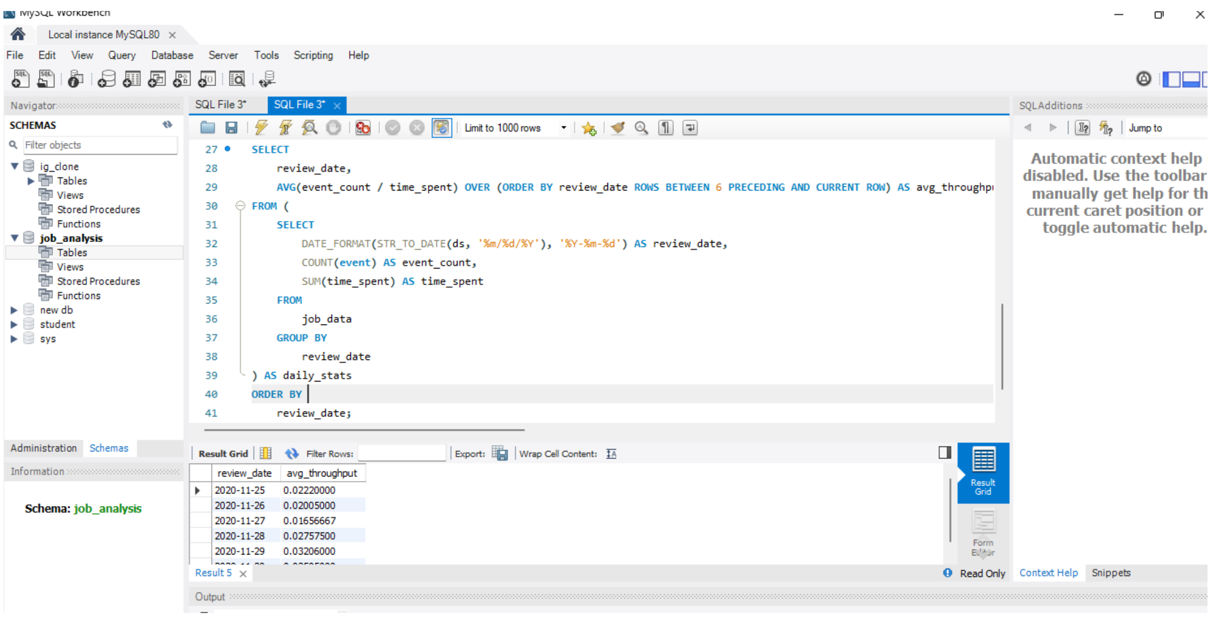
- **Jobs Reviewed Over Time**:



Result Grid | Filter Rows: | Export: | Wrap Cell Content:

review_date	review_hour	jobs_reviewed
2020-11-25	11	1
2020-11-26	11	1
2020-11-27	11	1
2020-11-28	11	2
2020-11-29	11	1
2020-11-30	11	2

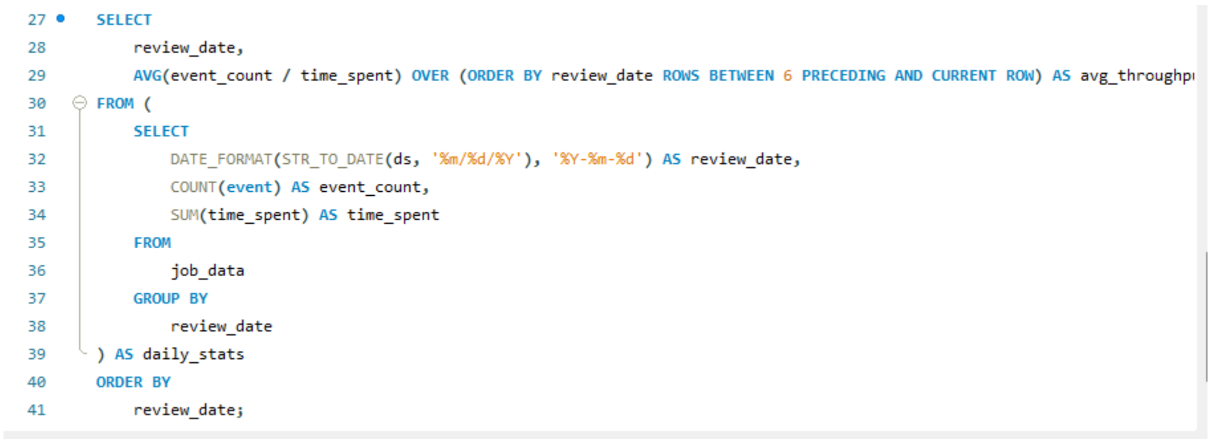
Task 2: Throughput Analysis



```sql
SELECT
    review_date,
    AVG(event_count / time_spent) OVER (ORDER BY review_date ROWS BETWEEN 6 PRECEDING AND CURRENT ROW) AS avg_throughpu
FROM (
    SELECT
        DATE_FORMAT(STR_TO_DATE(ds, '%m/%d/%Y'), '%Y-%m-%d') AS review_date,
        COUNT(event) AS event_count,
        SUM(time_spent) AS time_spent
    FROM
        job_data
    GROUP BY
        review_date
) AS daily_stats
ORDER BY
    review_date;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

review_date	avg_throughput
2020-11-25	0.02220000
2020-11-26	0.02005000
2020-11-27	0.01656667
2020-11-28	0.02757500
2020-11-29	0.03206000

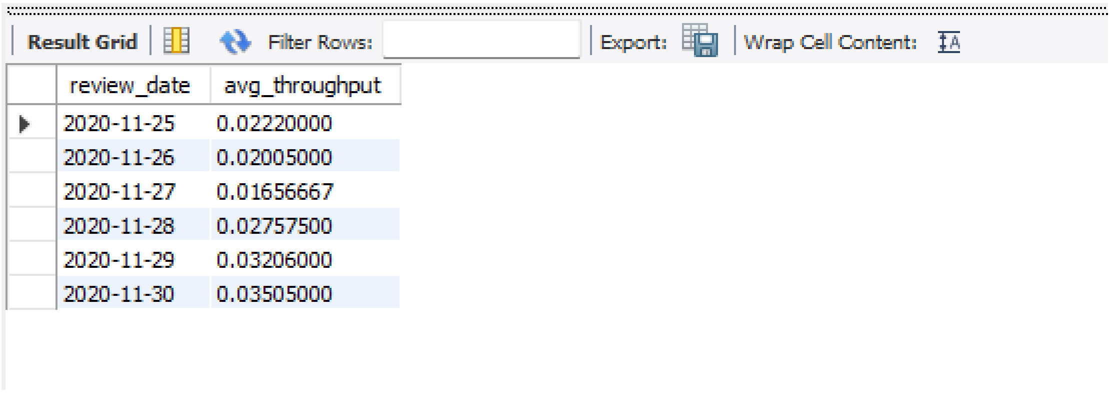


```sql
SELECT
    review_date,
    AVG(event_count / time_spent) OVER (ORDER BY review_date ROWS BETWEEN 6 PRECEDING AND CURRENT ROW) AS avg_throughp
FROM (
    SELECT
        DATE_FORMAT(STR_TO_DATE(ds, '%m/%d/%Y'), '%Y-%m-%d') AS review_date,
        COUNT(event) AS event_count,
        SUM(time_spent) AS time_spent
    FROM
        job_data
    GROUP BY
        review_date
) AS daily_stats
ORDER BY
    review_date;
```

```sql
SELECT
   review_date,
   AVG(event_count / time_spent) OVER (ORDER BY review_date ROWS BETWEEN 6 PRECEDING AND
CURRENT ROW) AS avg_throughput
FROM (
   SELECT
      DATE_FORMAT(STR_TO_DATE(ds, '%m/%d/%Y'), '%Y-%m-%d') AS review_date,
      COUNT(event) AS event_count,
      SUM(time_spent) AS time_spent
   FROM
      job_data
   GROUP BY
      review_date
) AS daily_stats
ORDER BY
   review_date;
```

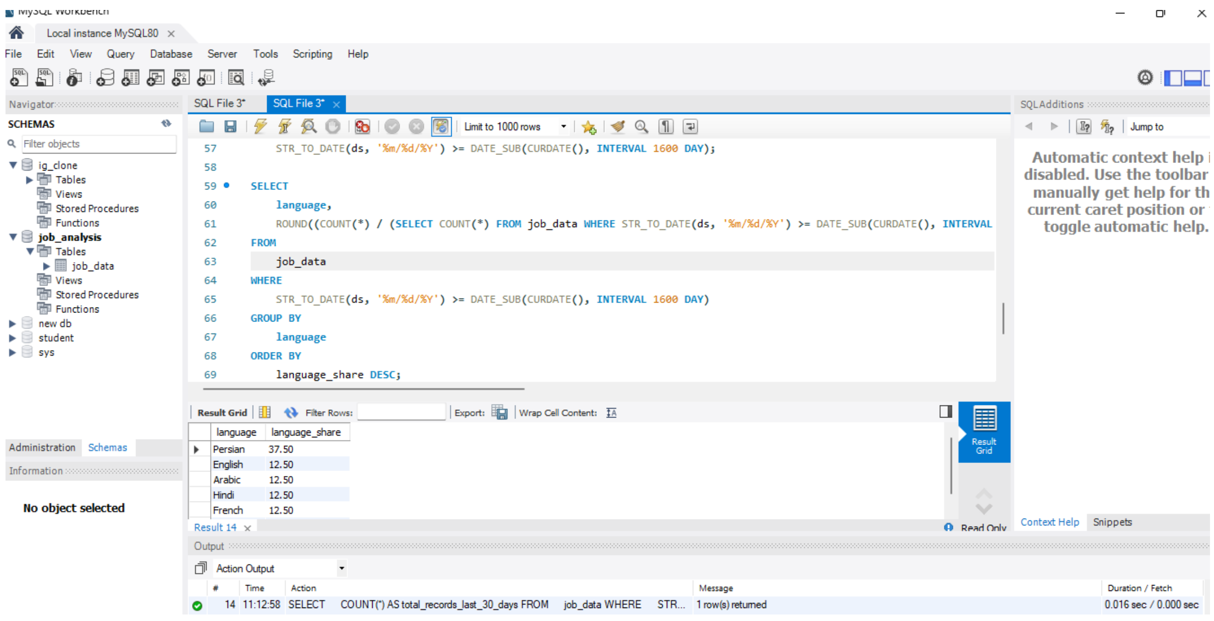
Insights

- **Throughput Analysis**:



	review_date	avg_throughput
▶	2020-11-25	0.02220000
	2020-11-26	0.02005000
	2020-11-27	0.01656667
	2020-11-28	0.02757500
	2020-11-29	0.03206000
	2020-11-30	0.03505000

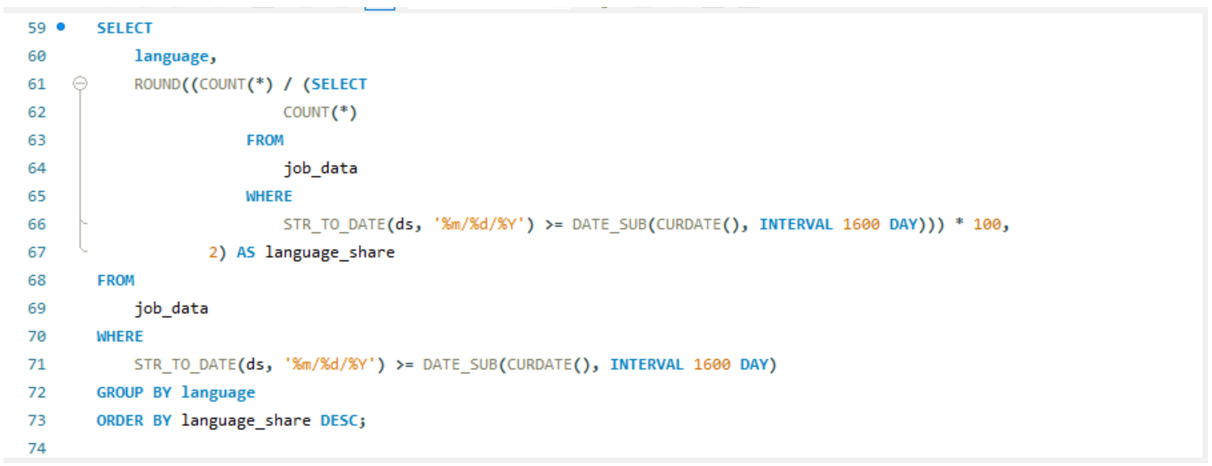
Task 3: Language Share Analysis



```
    STR_TO_DATE(ds, '%m/%d/%Y') >= DATE_SUB(CURDATE(), INTERVAL 1600 DAY);

SELECT
    language,
    ROUND((COUNT(*) / (SELECT COUNT(*) FROM job_data WHERE STR_TO_DATE(ds, '%m/%d/%Y') >= DATE_SUB(CURDATE(), INTERVAL
FROM
    job_data
WHERE
    STR_TO_DATE(ds, '%m/%d/%Y') >= DATE_SUB(CURDATE(), INTERVAL 1600 DAY)
GROUP BY
    language
ORDER BY
    language_share DESC;
```

	language	language_share
▶	Persian	37.50
	English	12.50
	Arabic	12.50
	Hindi	12.50
	French	12.50

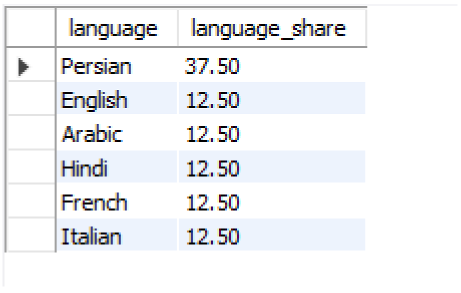
```
#   Time      Action                                                                  Message            Duration / Fetch
14  11:12:58  SELECT   COUNT(*) AS total_records_last_30_days FROM   job_data WHERE   STR...   1 row(s) returned  0.016 sec / 0.000 sec
```



```
SELECT
    language,
    ROUND((COUNT(*) / (SELECT
                COUNT(*)
            FROM
                job_data
            WHERE
                STR_TO_DATE(ds, '%m/%d/%Y') >= DATE_SUB(CURDATE(), INTERVAL 1600 DAY))) * 100,
            2) AS language_share
FROM
    job_data
WHERE
    STR_TO_DATE(ds, '%m/%d/%Y') >= DATE_SUB(CURDATE(), INTERVAL 1600 DAY)
GROUP BY language
ORDER BY language_share DESC;
```

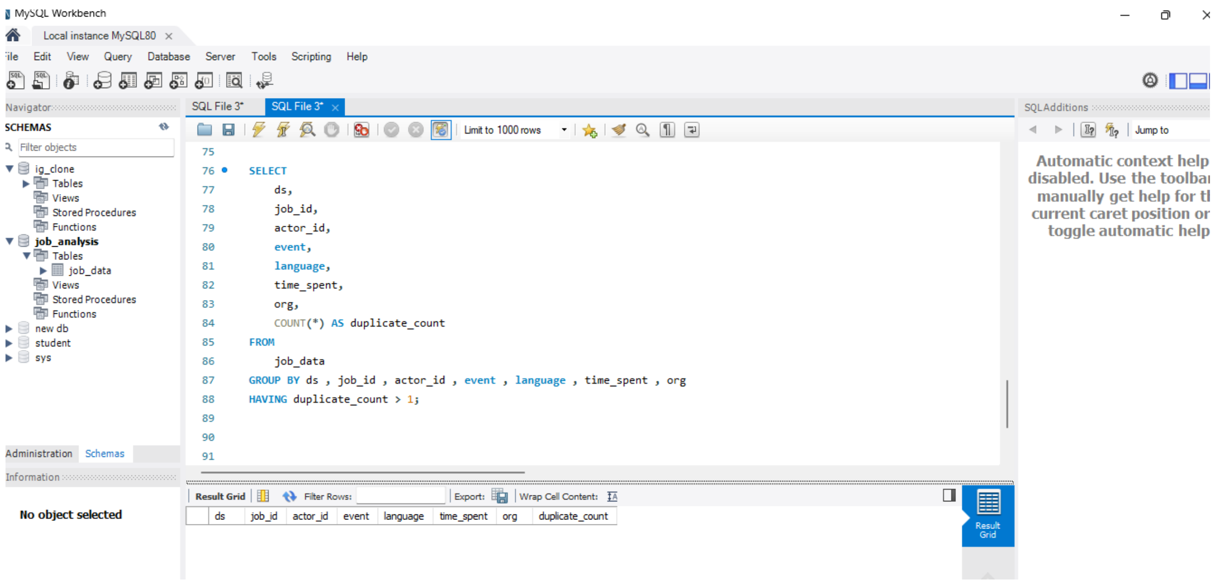
Insights

Language Share:



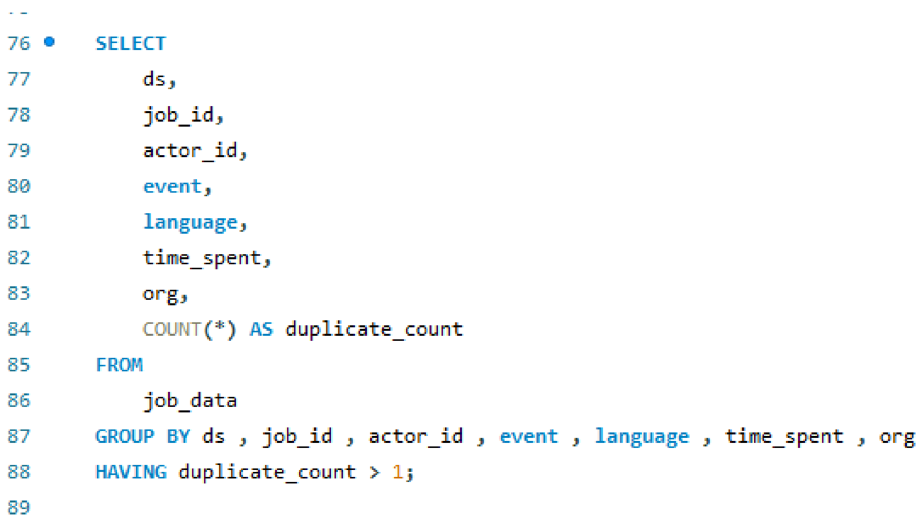
language	language_share
Persian	37.50
English	12.50
Arabic	12.50
Hindi	12.50
French	12.50
Italian	12.50

Task 4: Duplicate Rows Detection



```
SELECT
    ds,
    job_id,
    actor_id,
    event,
    language,
    time_spent,
    org,
    COUNT(*) AS duplicate_count
FROM
    job_data
GROUP BY ds , job_id , actor_id , event , language , time_spent , org
HAVING duplicate_count > 1;
```

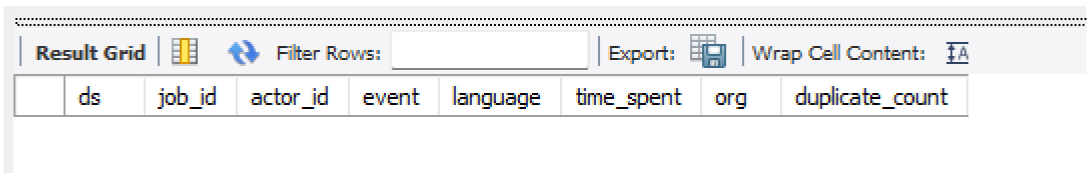
ds	job_id	actor_id	event	language	time_spent	org	duplicate_count



```
SELECT
    ds,
    job_id,
    actor_id,
    event,
    language,
    time_spent,
    org,
    COUNT(*) AS duplicate_count
FROM
    job_data
GROUP BY ds , job_id , actor_id , event , language , time_spent , org
HAVING duplicate_count > 1;
```

Insights:

Duplicate Detection



ds	job_id	actor_id	event	language	time_spent	org	duplicate_count

Insights

The analysis produced valuable insights:

- **Jobs Reviewed Over Time**: The hourly distribution of reviewed jobs provided visibility into peak and low activity periods.
- **Throughput Analysis**: The 7-day rolling average highlighted trends in job processing rates, allowing for the identification of consistent workflow patterns.
- **Language Share**: The percentage distribution of languages used in job reviews helped determine language popularity and could inform resource allocation for multilingual support.
- **Duplicate Detection**: No duplicates identified.