IS-5960: Masters Research Project

By

Team 10

Deepthi Boddu

Lakshmi Sruthi Bodipudi

Meghana Kakkireni

Narendra Reddy Kallem

Ramsailesh Reddy Kakarla

Seetha Rama Karthikeya Vempati

Saint Louis University

Instructor: Maria Weber

Database Integrity Checks

1.Ensuring Referential Integrity:

The database is structured with Job_Posting, Company, and Job_Applicant. To ensure valid foreign key relationships, we validate:

- Each job posting is linked to a valid company.
- Each job application references an existing job listing.

SQL Queries to Validate Referential Integrity

- Identify job postings linked to non-existent companies

```
SELECT COUNT(*) AS Orphaned_Jobs FROM Job_Posting
LEFT JOIN Company ON Job_Posting.company_id = Company.company_id
WHERE Company.company_id IS NULL;
```

- Identify job applications linked to missing job postings

```
SELECT COUNT(*) AS Orphaned_Applications FROM Job_Applicant
LEFT JOIN Job_Posting ON Job_Applicant.applied_job_id = Job_Posting.job_id
WHERE Job_Posting.job_id IS NULL;
```

2. Validating Field-Level Data Integrity

Beyond relational integrity, individual fields must conform to expected ranges and data types.

SQL Queries for Field-Level Integrity Checks

- Identify invalid salary values (should be positive)

```
SELECT COUNT(*) AS Invalid_Salaries FROM Job_Posting WHERE salary <= 0;
```

Result: 0, No invalid salaries found.

- Ensure job titles are present

```
SELECT COUNT(*) AS Missing_Job_Titles FROM Job_Posting WHERE title IS NULL OR title = '';
```

Result: 0, No missing job titles.

- Check if employer ratings fall within the valid range (0 to 5)

```
SELECT COUNT(*) AS Invalid_Ratings FROM Company WHERE rating < 0 OR rating > 5;
```

Result: 0, All employer ratings are valid.

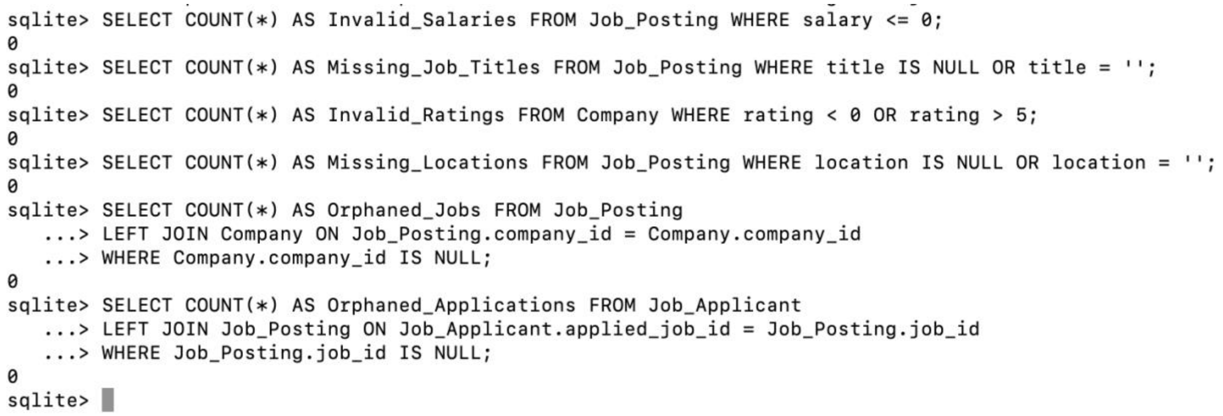
- Confirm that all job postings have an associated location

```
SELECT COUNT(*) AS Missing_Locations FROM Job_Posting WHERE location IS NULL OR location = '';
```

Result: 0, No missing job locations.

Validation Results & Screenshot

Below is a screenshot of the SQL queries executed and their results, verifying data integrity:



```
sqlite> SELECT COUNT(*) AS Invalid_Salaries FROM Job_Posting WHERE salary <= 0;
0
sqlite> SELECT COUNT(*) AS Missing_Job_Titles FROM Job_Posting WHERE title IS NULL OR title = '';
0
sqlite> SELECT COUNT(*) AS Invalid_Ratings FROM Company WHERE rating < 0 OR rating > 5;
0
sqlite> SELECT COUNT(*) AS Missing_Locations FROM Job_Posting WHERE location IS NULL OR location = '';
0
sqlite> SELECT COUNT(*) AS Orphaned_Jobs FROM Job_Posting
   ...> LEFT JOIN Company ON Job_Posting.company_id = Company.company_id
   ...> WHERE Company.company_id IS NULL;
0
sqlite> SELECT COUNT(*) AS Orphaned_Applications FROM Job_Applicant
   ...> LEFT JOIN Job_Posting ON Job_Applicant.applied_job_id = Job_Posting.job_id
   ...> WHERE Job_Posting.job_id IS NULL;
0
sqlite> 
```

All integrity checks returned 0, confirming the dataset is clean and valid.

DAX Queries Used in the Dashboard

1. Average Salary by Job Title

Average Salary = AVERAGE('Job_Posting'[salary])

2. Urgent Job Count

Urgent Jobs = CALCULATE(COUNTROWS('Job_Posting'), 'Job_Posting'[urgently_hiring] = TRUE())

3. Job Postings Over Time

Jobs Posted = COUNT('Job_Posting'[job_id])

5. Remote Work Model Distribution

Remote Jobs = CALCULATE(COUNTROWS('Job_Posting'), 'Job_Posting'[remote_work_model] = "Remote")

6. Featured Employers Count

Featured Employers = CALCULATE(COUNTROWS('Job_Posting'), 'Job_Posting'[featured_employer] = TRUE())

8. Total Unique Companies

```
Unique Companies = DISTINCTCOUNT('Job_Posting'[company_name])
```

10. Average Rating by Company

```
Average Rating = AVERAGE('Job_Posting'[rating])
```