

OPTIMIZING DATA MIGRATION AND QUERYING: A COMPREHENSIVE ANALYSIS

Presented By : Paras Dhiman (2021482)

**Indraprastha Institute Of Information
technology Delhi | 2024 | BDA (CSE557)**

AGENDA

- **Migration Process**
- **Key Queries**
- **Optimization Techniques**
- **Key Findings**
- **Conclusion**

MIGRATION PROCESS OVERVIEW

- **Initial Setup:** Migrated data from the legacy system to the new database.
- **Data Integrity:** Ensured that all records were transferred accurately without data loss.
- **Schema Mapping:** Mapped the old schema to the new schema to fit the new database structure.
- **Challenges:** Faced issues with data formats and incompatible data types.

KEY MIGRATION STEPS

- **Data Export:** Extracted data from the old database using export tools.
- **Schema Conversion:** Transformed the schema to match the new database structure.
- **Data Import:** Loaded data into the new database using bulk import methods.
- **Data Validation:** Performed checks to ensure data consistency post-migration.



COMMON QUERIES

4

Department-wise Student Count:

```
SELECT department_id, COUNT(student_id) AS total_students  
FROM enrollments  
GROUP BY department_id;
```

Fetching Student Information:

```
SELECT first_name, last_name, email  
FROM students;
```

Average Course Enrollment:

```
SELECT AVG(student_count) AS average_enrollment  
FROM enrollments;
```

ADVANCED QUERY EXAMPLES

5

Top-10 Courses by Enrollment:

```
SELECT c.course_id, c.course_name, COUNT(e.student_id) AS enrollment_count
FROM courses c
JOIN enrollments e ON c.course_id = e.course_id
GROUP BY c.course_id, c.course_name
ORDER BY enrollment_count DESC
LIMIT 10;
```

Instructor and Course Details:

```
SELECT i.first_name, i.last_name, c.course_name, c.department_id
FROM instructors i
JOIN course_instructors ci ON i.instructor_id = ci.instructor_id
JOIN courses c ON ci.course_id = c.course_id;
```

OPTIMIZATION TECHNIQUES

- **Indexing:**
 - Added indexes to frequently queried columns to speed up search queries.
- **Query Optimization:**
 - Rewrote complex queries to reduce execution time by avoiding redundant joins and using subqueries.
- **Partitioning:**
 - Implemented table partitioning to manage large datasets efficiently.
- **Caching:**
 - Used caching to store results of frequently executed queries.

KEY FINDINGS

- **Performance Improvements:**
 - Query execution times reduced by approximately 40% after applying optimizations.
- **Better Resource Allocation:**
 - Optimized queries allowed for faster decision-making and resource allocation, especially in student enrollment and department analysis.
- **Data Integrity:**
 - Maintained 100% data integrity post-migration, ensuring no loss of data.

CONCLUSION



- **Successful migration ensured seamless data transition from the old system to the new database.**
- **Implemented efficient querying strategies to improve data retrieval.**
- **Optimizations resulted in significant performance improvements.**
- **Future scope includes exploring advanced optimization techniques such as query parallelization.**

The background features three vertical stripes on the left: a wide pink stripe, a medium blue stripe, and a narrow beige stripe. The right side of the image is a light beige background with two rectangular areas of small, light pink dots in the top right and bottom right corners.

THANK YOU