Data Deduplication For Cloud Storage

DSA BOOTCAMP -2024

Computer Engineering & Information Technology

Submitted To:

Deepti Ma’am

(Bootcamp Trainer)

Submitted By:

Shivam Sharma (2247003)

Shivansh Sharma(2245729)

Shivani Sharma(2245918)

Shubham Kumar(2245915)

Sonu Kumar Das

Problem Statement and Objective

Problem Statement

As data volumes continue to grow exponentially, organizations face mounting challenges in managing and storing their data effectively. Traditional storage solutions often struggle to keep up with the increasing demand, leading to inefficient use of storage resources, higher costs, and potential data loss.

Objective

The primary objective of a data deduplication system is to optimize data storage by identifying and eliminating redundant data, reducing the overall storage footprint and improving the efficiency of data management processes. This, in turn, can lead to cost savings, improved backup and recovery capabilities, and enhanced data security.

Data Deduplication System

Data deduplication is a powerful technique that reduces data storage and network bandwidth requirements by eliminating redundant data. This is especially crucial in today's era of exponential data growth, where organizations must find efficient ways to manage and store vast amounts of information. A data deduplication system is a specialized software or hardware solution that identifies and removes duplicate data, optimizing storage and improving overall data management processes.

Architecture and Tools

Architecture

A typical data deduplication system consists of several key components, including a data ingestion module, a deduplication engine, a storage repository, and a management interface. The data ingestion module receives incoming data, the deduplication engine analyzes and identifies redundant data, and the storage repository stores the unique data segments.

Tools

In this project, we used Python with Flask for the web interface, **mysql.connector** for MySQL database integration, and **hashlib** to compute **SHA-256** hashes for file deduplication. The **os** and **shutil** modules manage file operations, while **Werkzeug** secures filenames and **jsonify** handles JSON responses for data exchange.

Deployment Models

Inline Deduplication: Deduplicates data before it's written to storage, saving space instantly but potentially introducing latency.
Post-Process Deduplication: Deduplicates data after writing, avoiding write latency but requiring extra temporary storage.
Choosing a Model: Depends on performance, storage needs, and whether real-time deduplication is required.

Code Snippets

Cloud_StorageDB

Limit to 1000 rows

```sql
CREATE DATABASE cloud_storage;

USE cloud_storage;

CREATE TABLE file_metadata (
    id INT AUTO_INCREMENT PRIMARY KEY,
    file_name VARCHAR(255) NOT NULL,
    display_name VARCHAR(255) NOT NULL,
    file_path VARCHAR(255) NOT NULL,
    file_hash VARCHAR(64) NOT NULL,
    file_size BIGINT NOT NULL,
    original_file_id INT DEFAULT NULL,
    is_duplicate BOOLEAN DEFAULT FALSE,
    FOREIGN KEY (original_file_id) REFERENCES file_metadata(id)
);


select * from file_metadata;

drop table file_metadata;
```

```python
import os
import hashlib
from flask import Flask, request, jsonify, render_template
import mysql.connector
from werkzeug.utils import secure_filename
import shutil

app = Flask(__name__)
UPLOAD_FOLDER = os.path.join(os.getcwd(), 'uploads')
os.makedirs(UPLOAD_FOLDER, exist_ok=True)

def connect_to_database():...

def compute_sha256(file_path):
    sha256_hash = hashlib.sha256()
    with open(file_path, "rb") as f:
        for byte_block in iter(lambda: f.read(4096), b""):
            sha256_hash.update(byte_block)
    return sha256_hash.hexdigest()

def format_size(size):
    if size == 0:
        return "0 Bytes"
    elif size < 1024:
        return f"{size} Bytes"
    elif size < 1024**2:
        return f"{size / 1024:.2f} KB"
    elif size < 1024**3:
        return f"{size / (1024**2):.2f} MB"
    else:
        return f"{size / (1024**3):.2f} GB"

@app.route('/')
def index():
    return render_template('index.html')
@app.route('/upload', methods=['POST'])
def upload_file():...
@app.route('/files', methods=['GET'])
def get_files():...
@app.route('/duplicates/<int:file_id>', methods=['GET'])
def get_duplicates(file_id):...
@app.route('/delete/<int:file_id>', methods=['DELETE'])
def delete_file(file_id):...
if __name__ == '__main__':
    app.run(debug=True)
```

Industrial Applications

Backup and Disaster Recovery

Data deduplication is widely used in backup and disaster recovery solutions to reduce the amount of data that needs to be stored and transmitted, improving backup performance and reducing storage costs.

Cloud Storage and File Sharing

Cloud storage and file-sharing services leverage data deduplication to optimize storage utilization and reduce costs, allowing users to store and share large amounts of data more efficiently.

Virtual Machine Images

Virtualized environments, such as data centers and cloud platforms, often use data deduplication to manage the storage of virtual machine images, which can contain significant amounts of redundant data.

Database Backups

Database administrators leverage data deduplication to optimize the storage and transmission of database backups, reducing the overall storage footprint and improving the efficiency of backup and recovery processes.

Future Developments and Conclusion

1

Artificial Intelligence and Machine Learning

As data deduplication systems continue to evolve, the integration of AI and ML algorithms will enable more advanced data analysis and optimization, leading to even higher deduplication ratios and improved storage efficiency.

2

Edge Computing and IoT

With the proliferation of IoT devices and the growing need for edge computing, data deduplication techniques will become increasingly important in managing the vast amounts of data generated at the edge, reducing the strain on centralized storage and network infrastructure.

3

Towards a Sustainable Future

As organizations strive to reduce their environmental impact, data deduplication will play a crucial role in minimizing the energy and resources required for data storage, contributing to more sustainable and eco-friendly data management practices.

In conclusion, data deduplication is a powerful technique that helps organizations optimize their data storage, reduce costs, and improve overall data management efficiency. As data volumes continue to grow and the demands on storage and network infrastructure increase, the importance of data deduplication will only continue to rise, paving the way for a more sustainable and efficient data-driven future.