SQL vs NO-SQL

Pushpak Sunil Rane

California State University Chico: Department of Mathematics and Statistics

MATH-608

DATA SCIENCE PROJECT

SQL vs NO-SQL

Pushpak Sunil Rane

California State University Chico: Department of Mathematics and Statistics

SQL – [Structured Query Language]

- SQL
- Oracle Database
- PostgreSQL
- IBM Db2
- Microsoft SQL Server
- MySQL
- SQLite
- MariaDB

NO-SQL – [Not Only SQL]

- CouchDB
- MongoDB
- Neo4j
- Cassandra
- Redis
- Elasticsearch
- Amazon DynamoDB

SQL vs NO-SQL

Pushpak Sunil Rane

California State University Chico: Department of Mathematics and Statistics

Feature	SQL	NoSQL
Data Model	Relational (tables)	Various (document, key-value, wide-column, etc.)
Schema	Fixed schema	Schema-less or dynamic schema
Scalability	Vertical scalability	Horizontal scalability
Query Language	Standardized (SQL)	Varies by database (e.g., MQL, CQL, etc)
Performance	Can be slower for large volumes of unstructured data	Optimized for fast read/write operations

SQL vs NO-SQL

Pushpak Sunil Rane

California State University Chico: Department of Mathematics and Statistics

SQL – [SQLite]

```python
import sqlite3
import pandas as pd

def store_data():
    sqlite_conn = sqlite3.connect('sqlite_grades.db')
    grade = sqlite_conn.cursor()

    # Create a table
    grade.execute('''
    CREATE TABLE IF NOT EXISTS grades (
        id INTEGER PRIMARY KEY,
        Name TEXT UNIQUE,
        MATH608 INTEGER,
        MATH615 INTEGER,
        CSCI605 INTEGER,
        ERTH600 INTEGER
    )
    ''')
```

```python
    # Insert multiple rows into the table
    try:
        grade.executemany('''INSERT INTO grades (Name, MATH608, MATH615, CSCI605, ERTH600)
                             VALUES (?, ?, ?, ?, ?) ''', grades_data)
        df = pd.read_sql_query('SELECT * FROM grades', sqlite_conn)
        print(df)
    except sqlite3.IntegrityError as e:
        print(f"Error occurred: {e}")

    # Commit changes and close the connection
    sqlite_conn.commit()
    sqlite_conn.close()
    print("Data has been created and stored in 'sqlite_grades.db'.")


store_data()
```

SQL vs NO-SQL

Pushpak Sunil Rane

California State University Chico: Department of Mathematics and Statistics

SQL – [SQLite]

```python
import os


def delete_database():
    database_path = 'sqlite_grades.db'

    if os.path.exists(database_path):
        os.remove(database_path)
        print(f"Database '{database_path}' has been deleted.")
    else:
        print(f"Database '{database_path}' does not exist.")

delete_database()
```

```python
def delete_row(name):
    # Connect to SQLite database
    sqlite_conn = sqlite3.connect('sqlite_grades.db')
    grade = sqlite_conn.cursor()

    # Delete row based on the name
    grade.execute('DELETE FROM grades WHERE Name = ?', (name,))

    # Commit changes and close the connection
    sqlite_conn.commit()
    sqlite_conn.close()
    print(f"Row with Name '{name}' has been deleted.")


delete_row("Govardhan Reddy Baddala")
```

SQL vs NO-SQL

Pushpak Sunil Rane

California State University Chico: Department of Mathematics and Statistics

SQL – [SQLite]

```python
def update_row(name, new_math608_score):
    # Connect to SQLite database
    sqlite_conn = sqlite3.connect('sqlite_grades.db')
    grade = sqlite_conn.cursor()

    # Update the MATH608 score for the specified name
    grade.execute('UPDATE grades SET MATH608 = ? WHERE Name = ?', (new_math608_score, name))

    # Commit changes and close the connection
    sqlite_conn.commit()
    sqlite_conn.close()
    print(f"Row with Name '{name}' has been updated with new MATH608 score: {new_math608_score}")

update_row("Govardhan Reddy Baddala", 10)
```

```python
import sqlite3
import pandas as pd
import matplotlib.pyplot as plt

sqlite_conn = sqlite3.connect('sqlite_grades.db')

query = "SELECT * FROM grades"
df = pd.read_sql_query(query, sqlite_conn)
sqlite_conn.close()

print("Retrieved Data:")
print(df)

fig, (ax1,ax2) = plt.subplots(1,2,figsize=(16, 6))
df.plot(kind = 'bar',x='Name', y='MATH608' , ax = ax1, color = 'green')
ax1.set_xlabel('Student Name',fontsize = 12)
ax1.set_ylabel('Grades',fontsize = 12)
ax1.set_title('Student Grades [MATH608]',fontsize = 16)
```

SQL vs NO-SQL

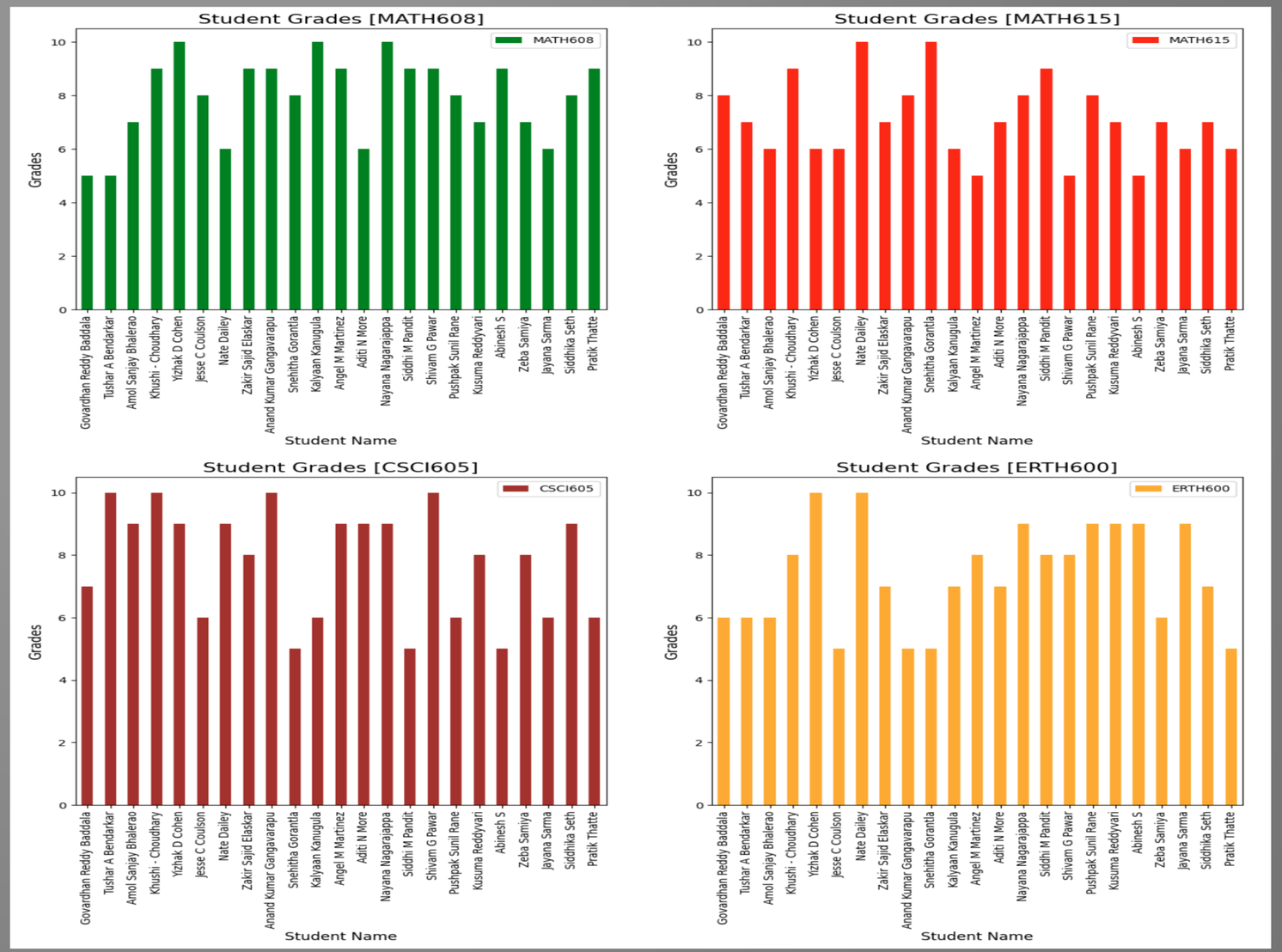
Pushpak Sunil Rane

California State University Chico: Department of Mathematics and Statistics

SQL – [SQLite]

Retrieved Data:

	id	Name	MATH608	MATH615	CSCI605	ERTH600
0	1	Govardhan Reddy Baddala	5	8	7	6
1	2	Tushar A Bendarkar	5	7	10	6
2	3	Amol Sanjay Bhalerao	7	6	9	6
3	4	Khushi - Choudhary	9	9	10	8
4	5	Yizhak D Cohen	10	6	9	10
5	6	Jesse C Coulson	8	6	6	5
6	7	Nate Dailey	6	10	9	10
7	8	Zakir Sajid Elaskar	9	7	8	7
8	9	Anand Kumar Gangavarapu	9	8	10	5
9	10	Snehitha Gorantla	8	10	5	5
10	11	Kalyaan Kanugula	10	6	6	7
11	12	Angel M Martinez	9	5	9	8
12	13	Aditi N More	6	7	9	7
13	14	Nayana Nagarajappa	10	8	9	9
14	15	Siddhi M Pandit	9	9	5	8
15	16	Shivam G Pawar	9	5	10	8
16	17	Pushpak Sunil Rane	8	8	6	9
17	18	Kusuma Reddyvari	7	7	8	9
18	19	Abinesh S	9	5	5	9
19	20	Zeba Samiya	7	7	8	6
20	21	Jayana Sarma	6	6	6	9
21	22	Siddhika Seth	8	7	9	7
22	23	Pratik Thatte	9	6	6	5



SQL vs NO-SQL

Pushpak Sunil Rane
California State University Chico: Department of Mathematics and Statistics

NO-SQL – [MongoDB]

```python
from pymongo import MongoClient,ASCENDING
from pymongo.errors import BulkWriteError

def store_data():
    student = MongoClient('mongodb://localhost:27017/')

    ds = student["grades_db"]

    grades_collection = ds["grades"]

    grades_collection.create_index([('Name', ASCENDING)], unique=True)

    grades_data = [
    {"Name": "Govardhan Reddy Baddala", "MATH608": 5, "MATH615": 8, "CSCI605": 7, "ERTH600": 6},
```

```python
    try:
        grades_collection.insert_many(grades_data, ordered=False)
        print("Data has been created and stored in MongoDB collection.")
    except BulkWriteError as e:
        print("Some documents were not inserted due to duplication: ",e.details)

    student.close()

store_data()
```

SQL vs NO-SQL

Pushpak Sunil Rane

California State University Chico: Department of Mathematics and Statistics

NO-SQL – [MongoDB]

```python
def fetch_data_from_mongodb():
    client = MongoClient('mongodb://localhost:27017/')

    db = client["grades_db"]
    collection = db["grades"]

    data = list(collection.find())

    client.close()

    return data

def print_data(data):
    df = pd.DataFrame(data)

    df.drop(columns=['_id'], inplace=True)

    print(df)

data = fetch_data_from_mongodb()
print_data(data)
```

```python
from pymongo import MongoClient


def delete_database():
    client = MongoClient('mongodb://localhost:27017/')

    client.drop_database('grades_db')

    client.close()
    print("Database 'grades_db' has been deleted.")

delete_database()
```

SQL vs NO-SQL

Pushpak Sunil Rane

California State University Chico: Department of Mathematics and Statistics

NO-SQL – [MongoDB]

```python
def update_record(name, new_math608_score):
    client = MongoClient('mongodb://localhost:27017/')

    db = client["grades_db"]
    collection = db["grades"]

    result = collection.update_one(
        {"Name": name},
        {"$set": {"MATH608": new_math608_score}}
    )

    if result.modified_count > 0:
        print(f"'{name}' has been updated with new MATH608 score: {new_math608_score}")
    else:
        print(f"'{name}' or no change was made.")

    client.close()

update_record("Govardhan Reddy Baddala", 10)
```

```python
def delete_record(name):
    client = MongoClient('mongodb://localhost:27017/')

    db = client["grades_db"]
    collection = db["grades"]

    result = collection.delete_one({"Name": name})
    print(f"Document with Name '{name}' has been deleted.")

    client.close()

delete_record("Govardhan Reddy Baddala")
```

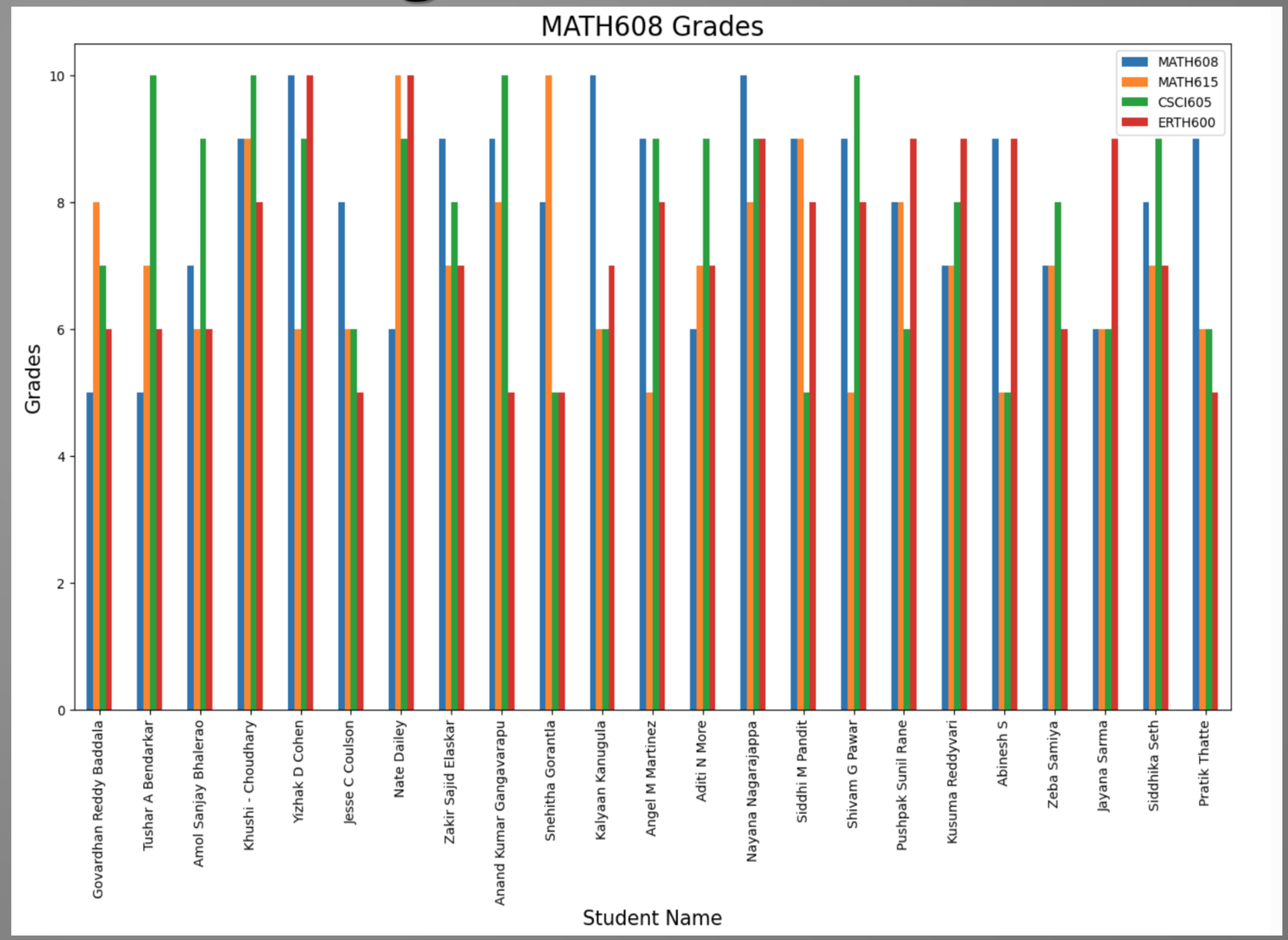
SQL vs NO-SQL

Pushpak Sunil Rane

California State University Chico: Department of Mathematics and Statistics

NO-SQL – [MongoDB]

	Name	MATH608	MATH615	CSCI605	ERTH600
0	Govardhan Reddy Baddala	5	8	7	6
1	Tushar A Bendarkar	5	7	10	6
2	Amol Sanjay Bhalerao	7	6	9	6
3	Khushi - Choudhary	9	9	10	8
4	Yizhak D Cohen	10	6	9	10
5	Jesse C Coulson	8	6	6	5
6	Nate Dailey	6	10	9	10
7	Zakir Sajid Elaskar	9	7	8	7
8	Anand Kumar Gangavarapu	9	8	10	5
9	Snehitha Gorantla	8	10	5	5
10	Kalyaan Kanugula	10	6	6	7
11	Angel M Martinez	9	5	9	8
12	Aditi N More	6	7	9	7
13	Nayana Nagarajappa	10	8	9	9
14	Siddhi M Pandit	9	9	5	8
15	Shivam G Pawar	9	5	10	8
16	Pushpak Sunil Rane	8	8	6	9
17	Kusuma Reddyvari	7	7	8	9
18	Abinesh S	9	5	5	9
19	Zeba Samiya	7	7	8	6
20	Jayana Sarma	6	6	6	9
21	Siddhika Seth	8	7	9	7
22	Pratik Thatte	9	6	6	5



SQL vs NO-SQL

Pushpak Sunil Rane

California State University Chico: Department of Mathematics and Statistics

THANK YOU