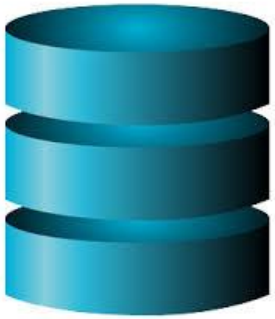
SIT32004
ICT Application Development

Mongo Database

Prof. Changbeom Choi

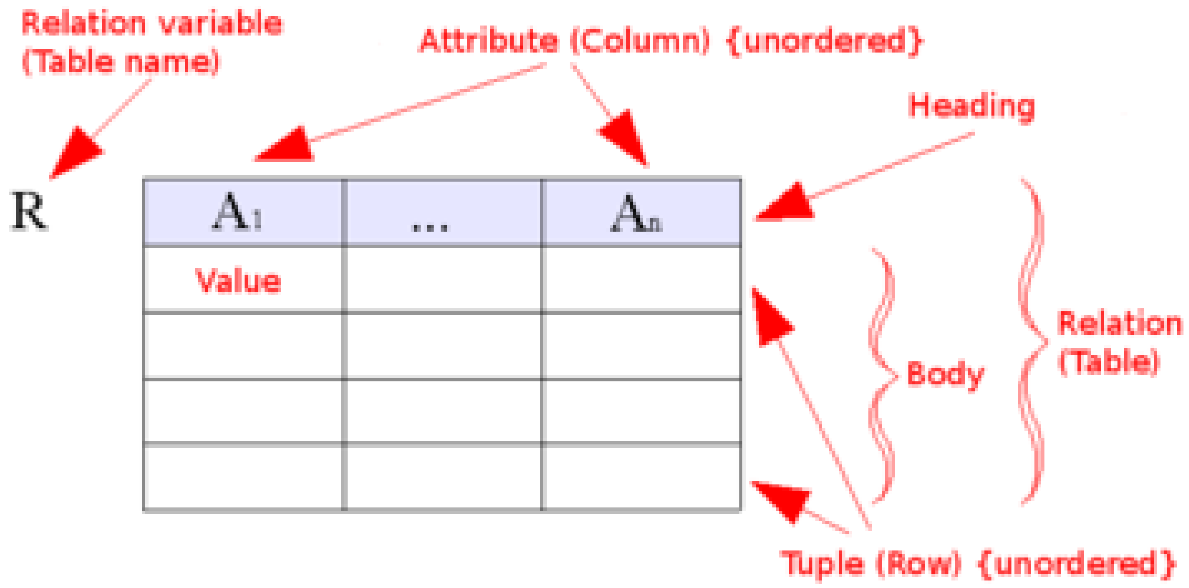
What is Database?[1]

- Definition
  - A database is an organized collection of data, generally stored and accessed electronically from a computer system
- Database Management System(DBMS)
  - DBMS is the software that interacts with end users, applications, and the database itself to capture and analyze the data.
  - The DBMS software additionally encompasses the core facilities provided to administer the database.
  - Well-known database: Relational databases
    - » These model data as rows and columns in a series of tables, and the vast majority use SQL for writing and querying data.



Relational Database (RDBMS) [2]

- Relation Model
  - An approach to managing data using a structure and language consistent with first-order predicate logic, where all data is represented in terms of tuples, grouped into relations
  - Provides a declarative method for specifying data and queries



Concepts of Relation Model, referenced from [2]

RDBMS, How it works? (1/3)

- Defining relation between data and data
  - "Key": records are linked together with "key"
  - "Join": Operation that combine multiple table into one

Student Name

ID	Family Name	First Name
2110001	Smith	John
2110002	Choi	Thea
2110003	Kim	Nana

Grade

ID	Grade
2110001	A+
2110002	A+
2110003	B+

Department

ID	Department
2110001	EECS
2110002	GEICT
2110003	ME

Join Table **Combining Table**

ID	Family Name	First Name	Grade	Department
2110001	Smith	John	A+	EECS
2110002	Choi	Thea	A+	GEICT
2110003	Kim	Nana	B+	ME

RDBMS, How it works? (2/3)

- Alter Table Example
  - Add new column

ID	Family Name	First Name	Grade	Department
2110001	Smith	John	A+	EECS
2110002	Choi	Thea	A+	GEICT
2110003	Kim	Nana	B+	ME

Adding new column to the table

ID	Family Name	First Name	Grade	Department	Internship
2110001	Smith	John	A+	EECS	Null
2110002	Choi	Thea	A+	GEICT	**20190325**
2110003	Kim	Nana	B+	ME	Null

System should maintain unnecessity space

RDBMS, How it works? (3/3)

- Alter Table Example
  - Add new table

Student Name

ID	Family Name	First Name
2110001	Smith	John
2110002	Choi	Thea
2110003	Kim	Nana

Grade

ID	Grade
2110001	A+
2110002	A+
2110003	B+

Department

ID	Department
2110001	EECS
2110002	GEICT
2110003	ME

Internship

ID	Internship
2110002	20190325

Want to know the name of the student who are participating internship?

Join table "Student Name" and "Internship"

ID	Family Name	First Name	Internship
2110001	Smith	John	Null
2110002	Choi	Thea	20190325
2110003	Kim	Nana	Null

Structured Query Language [3]

- The SQL language is subdivided into several language elements, including:
  - Clauses, which are constituent components of statements and queries.
  - Expressions, which can produce either scalar values, or tables consisting of columns and rows of data
  - Predicates, which specify conditions that can be evaluated to SQL three-valued logic (3VL) (true/false/unknown)
  - Queries, which retrieve the data based on specific criteria. This is an important element of SQL.
  - Statements, which may have a persistent effect on schemata and data, or may control transactions, program flow, connections, sessions, or diagnostics.
  - SQL statements also include the semicolon (";") statement terminator.
  - Insignificant whitespace is generally ignored in SQL statements and queries, making it easier to format SQL code for readability.

SQL Example

```
SELECT *
FROM Book
WHERE price > 100.00
ORDER BY title;
```

```
SELECT Book.title AS Title,
       count(*) AS Authors
 FROM  Book
 JOIN  Book_author
   ON  Book.isbn = Book_author.isbn
 GROUP BY Book.title;
```

Output Example

```
Title                              Authors
---------------------------------- -------
SQL Examples and Guide  4
The Joy of SQL                     1
An Introduction to SQL       2
Pitfalls of SQL                    1
```

NoSQL("non SQL" or "non relational")

- NoSQL DB[4]
  - Provides a mechanism for storage and retrieval of data that is modeled in means other than the tabular relations used in relational databases
  - NoSQL databases are increasingly used in big data and real-time web applications.

- Benefits of NoSQL[5]
  - When compared to relational databases, NoSQL databases are more scalable and provide superior performance, and their data model addresses several issues that the relational model is not designed to address:
    - » Geographically distributed architecture instead of expensive, monolithic architecture
    - » Large volumes of rapidly changing structured, semi-structured, and unstructured data
    - » Agile sprints, quick schema iteration, and frequent code pushes
    - » Object-oriented programming that is easy to use and flexible

NoSQL Types

- **Document databases**
  - Pair each key with a complex data structure known as a document.
  - Documents can contain many different key-value pairs, or key-array pairs, or even nested documents.
- **Graph** Stores
  - Used to store information about networks of data, such as social connections.
  - Graph stores include Neo4J and Giraph.
- **Key-value** Stores
  - Simplest NoSQL databases.
  - Every single item in the database is stored as an attribute name (or 'key'), together with its value.
- **Wide-column Stores**
  - Optimized for queries over large datasets, and store columns of data together, instead of rows.

MongoDB

- MongoDB is a cross-platform, document oriented database that provides
  - High performance.
  - High availability.
  - Easy scalability.
  - Works on concept of collection and document.
  - MongoDB stores data as the form of JavaScript Object Notation (JSON)

- JSON

```
{
  "name":"Jack B. Nimble",
  "at large": true,
  "grade":"A",
  "level":3,
  "format":{"type":"rect","width":1920,
"height":1080,"interlace":false, "framerate":24}
}
```

Object

Comparison between Python Dictionary and JSON

- JSON and Dictionary in Python are very similar to each other
  - Key-Value Pair
  - Value can be an object and may store various objects, such as list, dictionary and so on.

JSON

```
{
  "name":"Jack B. Nimble",
  "at large": true,
  "grade":"A",
  "level":3,
  "format":{"type":"rect","width":1920, "height":1080,"interlace":false,
"framerate":24}
}
```

Python Dictionary

```
{
 'name': 'Jack B. Nimble’,
 'at large': True,
 'grade': 'A’,
'level': 3,
'format': {'type': 'rect', 'width': 1920, 'height': 1080, 'interlace': False, 'framerate': 24}
}
```

Reference

[1] Database, https://en.wikipedia.org/wiki/Database

[2] Relational Model, https://en.wikipedia.org/wiki/Relational_model

[3] SQL Syntax, https://en.wikipedia.org/wiki/SQL_syntax

[4] NoSQL, https://en.wikipedia.org/wiki/NoSQL

[5] Benefits of NoSQL, https://www.mongodb.com/nosql-explained

SIT32004
ICT Application Development

Practice 05 Database

Prof. Changbeom Choi

Practice01: SQLite DB (1/6)

- Basic Structure

```
import sqlite3

# Connect to SQLite DB
conn = sqlite3.connect("test.db")

with conn:
    # Create Cursor from connection object
    cur = conn.cursor()

    # Put SQL Query

    # Reflect to Database
    conn.commit()
```

Practice01: SQLite DB (2/6)

- Create Table
  - SQL Query
    - CREATE TABLE **customer**(**id** integer primary key autoincrement, **name** text not null, **category** integer, **region** text);

Column Name

Table Name

```
import sqlite3

# Connect to SQLite DB
conn = sqlite3.connect("test.db")

with conn:
    # Create Cursor from connection object
    cur = conn.cursor()

    cur.execute("CREATE TABLE customer(id integer primary key autoincrement,
name text not null, category integer, region text);")


    # Reflect to Database
    conn.commit()
```

Practice01: SQLite DB (3/6)

- Insert Data to customer table
  - SQL Query
    - INSERT INTO customer(name, category, region) VALUES (‘cbchoi’, 1, ‘Daejeon’)

```
import sqlite3

# Connect to SQLite DB
conn = sqlite3.connect("test.db")

with conn:
    # Create Cursor from connection object
    cur = conn.cursor()

    cur.execute("INSERT INTO customer(name, category, region) values ('cbchoi',
1, 'Daejeon');")

    # Reflect to Database
    conn.commit()
```

Practice01: SQLite DB (4/6)

- Retrieve Data to customer table
  - SQL Query
    - SELECT * FROM customer
    - SELECT name FROM customer WHERE category = 1

```
import sqlite3

# Connect to SQLite DB
conn = sqlite3.connect("test.db")

with conn:
    # Create Cursor from connection object
    cur = conn.cursor()

    # SQL Query
    cur.execute("SELECT name from customer WHERE category = 1;")

    for row in cur.fetchall():
        print(row)

    conn.commit()
```

Practice01: SQLite DB (5/6)

- Insert List to the SQLite DB
  - ? Syntax
    - » sql = "insert into customer (name,category, region) values (?,?,?)“
  - Usage
    - » infoList = [(‘t.choi’, 1, ‘Seoul'), (‘J.Kim', 2, ‘Busan'), (‘M.Lee’, 2, ‘Daejeon')]
    - » cur.executemany(sql,infoList)

```
import sqlite3

# Connect to SQLite DB
conn = sqlite3.connect("test.db")

with conn:
    # Create Cursor from connection object
    cur = conn.cursor()
    sql = "insert into customer (name,category, region) values (?,?,?)"
    infoList = [('t.choi', 1, 'Seoul'), ('J.Kim', 2, 'Busan'), ('M.Lee', 2, 
'Daejeon')]
    cur.executemany(sql,infoList)

    cur.execute('select* from customer')
    [print(row) for row in cur.fetchall()]

    conn.commit()
```

Practice01: SQLite DB (6/6)

- Remove data from SQLite DB
  - SQL Query
    - » DELETE FROM *table_name* WHERE *condition*;

```
import sqlite3

# Connect to SQLite DB
conn = sqlite3.connect("test.db")

with conn:
    # Create Cursor from connection object
    cur = conn.cursor()
    cur.execute("DELETE FROM customer WHERE region = 'Daejeon'")
    cur.execute("INSERT INTO customer(name, category, region) values ('cbchoi',
1, 'Daejeon')")

    cur.execute('select* from customer')
    [print(row) for row in cur.fetchall()]

    conn.commit()
```

Practice02: Table Join (1/2)

- Assume that you have three tables

Student Name

ID	Family Name	First Name
2110001	Smith	John
2110002	Choi	Thea
2110003	Kim	Nana

Grade

ID	Grade
2110001	A+
2110002	A+
2110003	B+

Department

ID	Department
2110001	EECS
2110002	GEICT
2110003	ME

- – Create three tables
  - » `CREATE TABLE StuName(id integer primary key, f.name text not null, l.name text not null)`
  - » `CREATE TABLE Grade(id integer primary key, grade text not null, l.name text not null)`
  - » `CREATE TABLE Department(id integer primary key, dept text not null)`
- – Insert data to the tables

Practice02: Table Join (2/2)

- Table Join
  - Syntax: SELECT * FROM tbl1, tbl2 where tbl1.col = tbl2.col

```
SELECT *
FROM StuName, Grade, Department
WHERE StuName.id = Grade.id AND Grade.id = Deaprtment.id
```

  - Use following functions

```
def create_tbl(conn):
    conn.execute("CREATE TABLE StuName(id integer primary key, fname text not null, lname text not null)")
    conn.execute("CREATE TABLE Grade(id integer primary key, grade text not null)")
    conn.execute("CREATE TABLE Department(id integer primary key, dept text not null)")

def insert_to_tbl(conn):
            conn.execute("INSERT INTO StuName(id, fname, lname) VALUES (2110001, 'Smith', 'John')")
            conn.execute("INSERT INTO StuName(id, fname, lname) VALUES (2110002, 'Choi', 'Thea')")
            conn.execute("INSERT INTO StuName(id, fname, lname) VALUES (2110003, 'Kim', 'Nana')")

            conn.execute("INSERT INTO Grade(id, grade) VALUES (2110001, 'A+')")
            conn.execute("INSERT INTO Grade(id, grade) VALUES (2110002, 'A+')")
            conn.execute("INSERT INTO Grade(id, grade) VALUES (2110003, 'B+')")

            conn.execute("INSERT INTO Department(id, dept) VALUES (2110001, 'EECS')")
            conn.execute("INSERT INTO Department(id, dept) VALUES (2110002, 'GEICT')")
            conn.execute("INSERT INTO Department(id, dept) VALUES (2110003, 'ME')")
```

Installing MongoDB

- You may refer the following Site
  - https://docs.mongodb.com/manual/administration/install-community/
  - Before starting mongoDB, you should make db folder
    - mkdir c:\data\db
    - "C:\Program Files\MongoDB\Server\4.0\bin\mongod.exe" --dbpath="c:\data\db"

- You may use online MongoDB hosting
  - https://studio3t.com/whats-new/cheap-free-mongodb-hosting/

- Python Setup
  - pip3 install pymongo or pip install pymongo

Practice03: MongoDB (1/5)

- Basic Structure

```
import pymongo

conn = pymongo.MongoClient('localhost', 27017)
db = conn.get_database('mongo_test')
collection = db.get_collection('customer')
```

If there is no ‘mongo_test’ database or ‘customer’ collection, the MongoDB will generate one.

Practice03: MongoDB (2/5)

- Insert data to MongoDB

```
import pymongo

conn = pymongo.MongoClient('localhost', 27017)
db = conn.get_database('mongo_test')
collection = db.get_collection('customer')

collection.insert_one({"name":"cbchoi", "category":1,
"region":'Daejeon'})

results = collection.find()

[print(result) for result in results]
```

Practice03: MongoDB (3/5)

- Retrieve Data from MongoDB

```
import pymongo

conn = pymongo.MongoClient('localhost', 27017)
db = conn.get_database('mongo_test')
collection = db.get_collection('customer')

results = collection.find({"region":'Pohang'})

[print(result) for result in results]
```

Practice03: MongoDB (4/5)

- Update/Replace data to MongoDB

```
import pymongo

conn = pymongo.MongoClient('localhost', 27017)
db = conn.get_database('mongo_test')
collection = db.get_collection('customer')

collection.replace_one({'name':'cbchoi1'}, {'name':'cbchoi1',
'category':2, 'region':'Daejeon'})

results = collection.find()
[print(result) for result in results]
```

Practice03: MongoDB (5/5)

- Remove data from MongoDB

```
import pymongo

conn = pymongo.MongoClient('localhost', 27017)
db = conn.get_database('mongo_test')
collection = db.get_collection('customer')

collection.delete_one({'name':'cbchoi1'})

results = collection.find()
[print(result) for result in results]
```

Homework05: RUNNING A STORE, REVISITED

- Suppose that your neighbor opened a vegetable and fruit store. In order to provide a better service to customers, he has decided to use a computer program for his business. He offered you a summer internship at his store to develop this program, and you have happily accepted his offer. This program should handle three kinds of transactions: **selling items**, **listing the current stock**, and **reporting all sales done during the day**.
- Your task is to modify the given program to use SQLite and MongoDB instead of text file.
  - You should submit two program:
    - » Store Management Program with SQLite
      - Design a Stock Table
    - » Store Management Program with MongoDB
      - Design a Stock Schema