



មជ្ឈមណ្ឌលកូរ៉េ សហវិវ អេច អ ឌី
Korea Software HRD Center



មូលនិធិ សហវិវ កូរ៉េ សម្រាប់ជំនួយជាសកល
Foundation for Korea Software Global Aid

ប្រធានបទ៖

Database Concept

អ្នកណែនាំ៖ លោក កៀន ដូច
លោក ឡេង ហុងម៉េង
លោក ស៊ាប ឡុងឌី



ប្រធានបទ៖

Database Concept

សមាជិក៖

១. ចាន់ សីហា

២. នុត វីរៈ

៣. យិន វ៉ាន់ធី

៤. ហេង ស្រីតូច

៥. មាន គន្ធា

៦. សេង ឈុនយ៉ាង

៧. ជា បញ្ញា

ថ្នាក់ ភ្នំពេញ

Group 1

មាតិកា

១. ផែនការយល់ពី Database Concept
២. ផែនការយល់ពីភាពខុសគ្នារវាង File and Database
៣. ផែនការយល់ពី Database Modeling (Conceptual/ Logical/ Physical Modeling)
៤. ផែនការយល់ពី JDBC Driver setting
៥. ផែនការយល់ពី Basic procedure of JDBC
៦. ផែនការយល់ពី Connection
៧. ឯកសារយោង

១. ស្វែងយល់ពី **Database Concept**

អ្វីទៅជា **Database Concept**?

Database concept គឺ refer ទៅលើ fundamental Principles ដែលវាធ្វើការរៀបចំទិន្នន័យនិងការគ្រប់គ្រងទិន្នន័យនៅក្នុង Database ។

Database conceptមានដូចជា:

- Database Schema
- Data Constraints
- Data Dictionary

១. ស្វែងយល់ពី **Database Concept**

អ្វីទៅជា **Database Concept**?(ត)

- Database Instance
- Query
- Data manipulation
- Data Engine

១. ស្វែងយល់ពី Database Concept

អ្វីទៅជា Database Concept?(ត)

- **Database Schema:** ជាការDesign databaseដែលយើងប្រើសម្រាប់តំណាងអោយ Structure និងប្រភេទនៃDataត្រូវបានstoreក្នុងrows និង columns, constraints, relationships between the tables។
- **Data Constraints:** ជា rule ឬជា condition ដែលកំណត់ទៅលើ column របស់ table។
- **Data Dictionary:** វាជាdatabase schemaដែលមានប្រភេទនៃconstraintsខុសគ្នា នៅក្នុងtableត្រូវបានរក្សាទុកដោយDBMS នៅក្នុងdictionaryដោយត្រូវបានគេស្គាល់ថាវាជា ទិន្នន័យ។

១. ស្វែងយល់ពី Database Concept

អ្វីទៅជា Database Concept?(ត)

- **Data Instance:** ជាការsetនៃmemory structureនិងbackgroundនៃការ processesដែលត្រូវបានប្រើសម្រាប់access database files ។
- **Query:** គឺជាការស្នើសុំទិន្នន័យយកមកប្រើប្រាស់។
- **Data manipulation:** ងាយស្រួលសម្រាប់កំណត់ទិន្នន័យដែលមានដូចជា Inert, Update និងDelete ។
- **Data Engine:** វាជាcomponentមួយដែលយើងប្រើសម្រាប់បង្កើតនិងគ្រប់គ្រងទិន្នន័យផ្សេងៗ។

២. ស្វែងយល់ពីភាពខុសគ្នារវាង **File and Database**

❖ អ្វីទៅជា **File** ?

File គឺជាការ method មួយសម្រាប់ store and organize data or file ហើយវា ផ្ទុកនៅក្នុង Hard-Disk ឬ optical disc ។

❖ អ្វីទៅជា **Database**?

Database គឺជាបណ្តុំទិន្នន័យដែលបានរៀបចំឡើងដើម្បីគ្រប់គ្រង រក្សាទុកនិងធ្វើការដោយ ប្រើអេឡិចត្រូនិកពី computer system ។

២. ស្វែងយល់ពីភាពខុសគ្នារវាង File and Database(ត)

❖ ភាពខុសគ្នារវាង File system and DBMS:

File System	DBMS
សម្រាប់គ្រប់គ្រង file នៅក្នុង storage medium	ជាsoftwareសម្រាប់គ្រប់គ្រង database
វាអាចកើតមាន data redundant	វាមិនអាចកើតមាន data redundant

២. ស្វែងយល់ពីភាពខុសគ្នារវាង File and Database(ត)

❖ ភាពខុសគ្នារវាង File system and DBMS:(ត)

វាមិនធ្វើការ backup and recovery dataទេ ប្រសិនបើ lost file	វាផ្តល់ tools សម្រាប់ backup and recovery data
វាមានភាពងាយស្រួលបើប្រៀបធៀបទៅនឹង DMBS	DBMS វាមានភាពពិបាកបើធៀបទៅនឹង file system
វាមិនសូវមានសុវត្ថិភាពដូច DMBS	វាមានសុវត្ថិភាពជាង file system

២. ស្វែងយល់ពីភាពខុសគ្នារវាង **File and Database(ត)**

❖ ភាពខុសគ្នារវាង **File system and DBMS:(ត)**

វាមិនមាន data independency	DBMS មាន data independency 2 (1. Logical Data Independency 2. Physical Data Independency)
វាអាចអនុញ្ញាតអោយតែ 1 user access dataបានក្នុងពេលដូចគ្នា	វាអាចអនុញ្ញាតអោយច្រើន user access dataបានក្នុងពេលដូចគ្នា
វាមានភាពលំបាកនៅក្នុងការ share data	DBMS វាមានភាពងាយស្រួលក្នុងការ share data

២. ស្វែងយល់ពីភាពខុសគ្នារវាង **File and Database** (តូចបំ)

❖ ភាពខុសគ្នារវាង **File system and DBMS:(តូចបំ)**

Integrity Constraints វាពិបាកក្នុងការ implement data	Integrity Constraints វាមានភាពងាយ ស្រួល implement data
វាអាច access data ទៅក្នុង file, user, require attributes such as file name, file location	DBMS វាមិនមានការ access បែបនេះទេ។

៣. ស្វែងយល់ពី **Database Modeling**

❖ អ្វីទៅជា database Modeling ?






Database modeling គឺជា process នៃការបង្កើត Structure ដើម្បីបង្ហាញពី Communicate Connection relationship ពី point data និង Structure។

database Modeling មាន 3 ៖

- ❑ Conceptual
- ❑ Logical
- ❑ Physical Modeling

៣. ស្វែងយល់ពី Database Modeling

❖ Information Engineering (IE) Notation

One and only one	
many	
Zero or many	
One or many	
One	

៣. ស្វែងយល់ពី Database Modeling (ត)

❖ Conceptual data modeling

Conceptual គឺជា highly abstract មិនត្រូវការ detail ងាយស្រួលយល់ព្រោះវាមានតែ entities និង relationship ។ conceptual មានតែ entities និង relationship ។ អត់មាន attributes និង Primary key ទេ។

Example



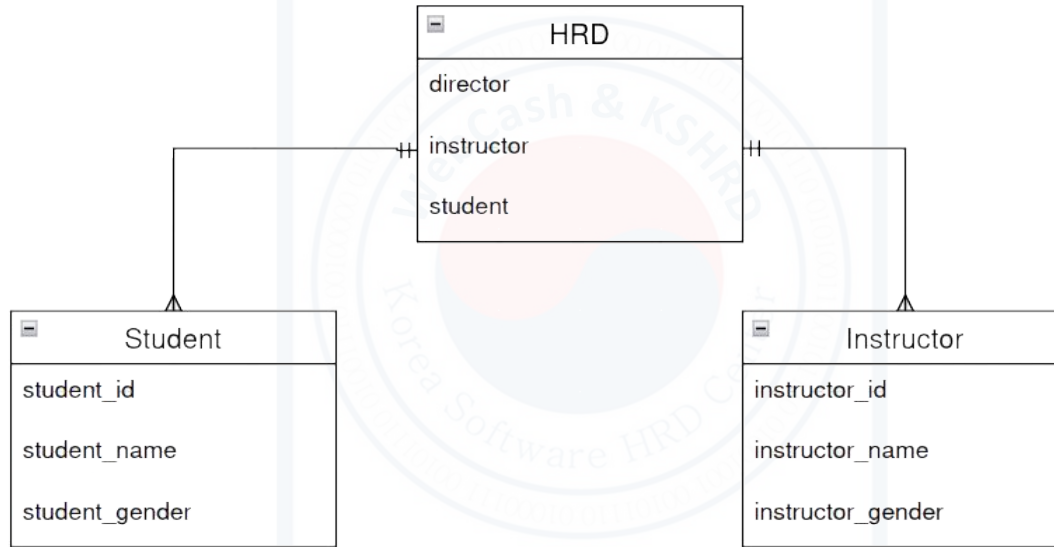
៣. ស្វែងយល់ពី Database Modeling (ត)

❖ Logical data modeling

Logical គឺជា representation នៃ data និង relationships វាមាន attributes, primary key, foreign key។

៣. ស្វែងយល់ពី Database Modeling (ត)

Example



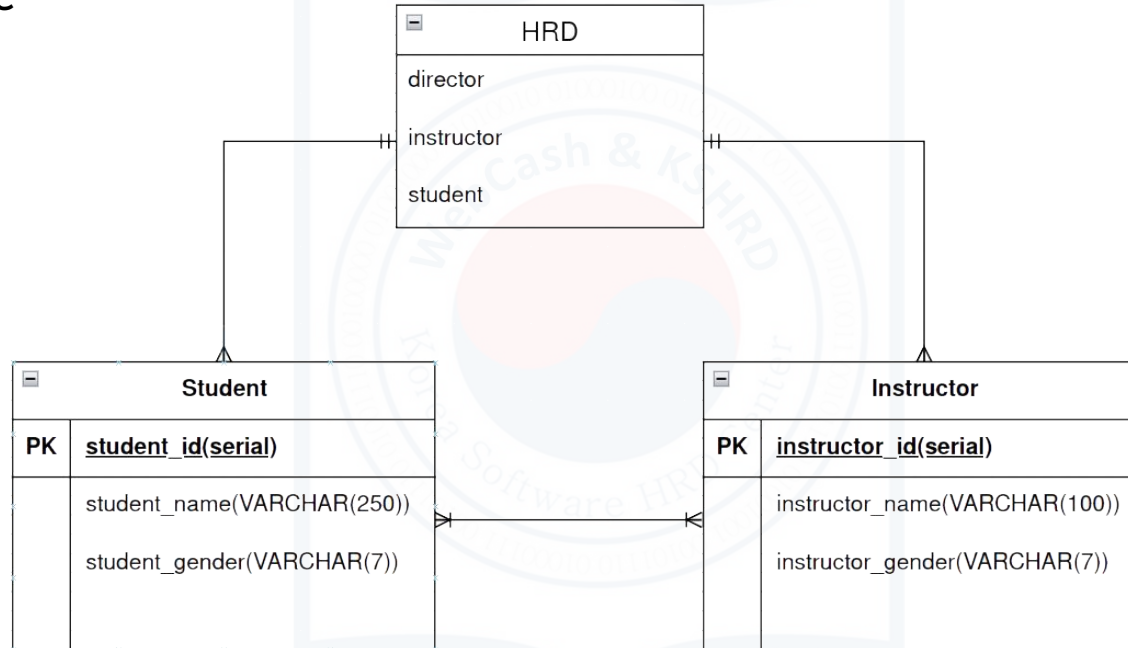
៣. ស្វែងយល់ពី Database Modeling (ត)

❖ តើអ្វីជា Physical Modeling ?

Physical modeling វាមានលក្ខណៈលើសពី logical modeling បន្តិចដែលយើងអាចបន្ថែម Data type និង Constraint ទៅលើ attributes របស់ table។

៣. ស្វែងយល់ពី Database Modeling (ត)

Example



៤. ស្វែងយល់ពី **JDBC Driver setting**

❖ **What is JDBC Driver?**

JDBC Driver (Java Database Connectivity) Driver Setting គឺជាដំណើរការនៃការ config java environment ដើម្បីប្រើប្រាស់ JDBC Driver ជាកំណត់ត្រាណាមួយសម្រាប់ Connect Java Application ទៅកាន់ Database។

៤. ស្វែងយល់ពី JDBC Driver setting (ត)

❖ What is JDBC Driver? (ត)

ដំណើរការនេះត្រូវការ Step មួយចំនួនដូចជា ៖

- Download JDBC Driver : ចូលទៅកាន់ website database ដែលអ្នកចង់ប្រើប្រាស់ (MySQL, PostgreSQL, Oracle...) ដើម្បី download JDBC Driver។
- Include JDBC Driver In Project : បន្ទាប់ពី download JDBC Driver ហើយ add JAR file ទៅកាន់ project directory។
-

៤. ស្វែងយល់ពី **JDBC Driver setting** (ត)

- Set Class Path : set class path ចូលទៅក្នុង java application ដោយប្រើប្រាស់នូវ -classpath ឬ -cp option ជាមួយ javac និង java commands។
- Load JDBC In The Code : ដោយប្រើប្រាស់នូវ Class.forName() method

```
Class.forName("com.postgresql.cj.jdbc.Driver");
```

៤. ស្វែងយល់ពី **JDBC Driver setting** (តិចម្ល៉ៃ)

- Establish Connection : បន្ទាប់ពី Loading JDBC Driver យើងអាច connect ទៅកាន់ database របស់យើងដោយប្រើប្រាស់ DriverManager.getConnection() method.
- Handle Exception : យើងអាចធ្វើការគ្រប់គ្រងការ error (handle exception) ដែលអាចនឹងកើតមានកំឡុងពេលធ្វើ operation បានដូចជា

ClassNotFoundException and SQLException

៥. ស្វែងយល់ពី **Basic Procedure of JDBC**

ក្នុងចំណុចនេះយើងនឹងស្វែងយល់ពីជំហាននៃការ Connect Java ជាមួយ Database។

ក្នុងការ connect នេះផងដែរ គឺមាន 6 ជំហាន៖

1. **Load the JDBC driver**

ក្នុងជំហាននេះយើងត្រូវប្រើប្រាស់ `Class.forName()`។ នោះយើងអាច load driver ទៅក្នុង memory ដូច្នេះវាអាច interact ជាមួយ database បាន។

```
Class.forName("com.postgresql.jdbc.Driver");
```


៥. ស្វែងយល់ពី **Basic Procedure of JDBC (ត)**

2. Establish connection

ក្នុងជំហាននេះយើងត្រូវ establish connection ទៅនឹង database ដោយប្រើ `Driver.Manager.getConnection()` method។ ហើយយើងត្រូវតែដាក់ URL របស់ database ដែលមាន username និង password។

```
Connection connection =  
DriverManager.getConnection("jdbc:postgresql://localhost:3306/m  
ydatabase", "username", "password");
```

៥. ស្វែងយល់ពី **Basic Procedure of JDBC (ត)**

3. Create statement

សរសេរ statements ដើម្បីធ្វើការ queries។

```
Statement statement = connection.createStatement();  
    PreparedStatement preparedStatement =  
    connection.prepareStatement("SELECT * FROM users WHERE id = ?");  
CallableStatement callableStatement = connection.prepareCall("{call  
    get_user(?)})");
```

៥. ស្វែងយល់ពី **Basic Procedure of JDBC (ត)**

4. **Execute Query/Update**

ក្នុងជំហាននេះយើងប្រើ `executeQuery()` ដើម្បី `execute` នូវ query statement របស់យើង (SELECT query) និង `executeUpdate()` method ដើម្បី `execute` INSERT, UPDATE និង DELETE queries។

```
ResultSet resultSet = statement.executeQuery("SELECT * FROM  
users");  
  
int rowsAffected = statement.executeUpdate("INSERT INTO users  
(name, age) VALUES ('John', 30)");
```

៥. ស្វ័យយល់ពី **Basic Procedure of JDBC (ត)**

5. Process Result

ប្រសិនបើយើង execute SELECT query នោះយើងអាច process the Result object ដើម្បី retrieve ទិន្នន័យ។

```
while(resultSet.next()) {  
    String name = resultSet.getString("name");  
    int age = resultSet.getInt("age");  
    System.out.println("Name: " + name + ", Age: " + age);}
```

៥. ស្វែងយល់ពី **Basic Procedure of JDBC (តទម៌)**

6. Close Resources

បន្ទាប់ពីធ្វើការរួចរាល់ហើយយើងត្រូវធ្វើការបិទ resources វិញដូចជា Connection, Statement and Result object ដើម្បីផ្តាច់ database និង JDBC resource។

```
resultSet.close();  
statement.close();  
connection.close();
```

៦. ស្វ័យប្រវត្តិ **Connection**

JDBC មានវិធីសាស្ត្រផ្សេងៗក្នុងការ establish database connection។ វិធីសាស្ត្រចាស់គឺធ្វើការ register driver ជាមួយ DriverManager។ ដោយវិធីសាស្ត្រថ្មីត្រូវបាន introduced នៅក្នុង JDBC 4.0 ដោយធ្វើការ load driver ដោយ automatic។

៦. ស្វ័យប្រវត្តិ Connection (ត)

❖ Old Way (Explicit Driver Registration)

```
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.SQLException;

public class OldConnectionExample {
    public static void main(String[] args) {
        // JDBC URL for MySQL database
        String url = "jdbc:mysql://localhost:3306/mydatabase";
        String username = "your_username";
        String password = "your_password";
        // Initialize connection object
        Connection connection = null;
```

៦. ស្វ័យយល់ពី **Connection** (ត)

❖ **Old Way (Explicit Driver Registration) (តិចប៉ុ)**

```
try {  
    // Explicitly register the JDBC driver  
    Class.forName("com.mysql.cj.jdbc.Driver");  
    // Establish connection  
    connection = DriverManager.getConnection(url, username, password);  
    System.out.println("Connected to the database!");  
    // Perform database operations here  
} catch (ClassNotFoundException e) {  
    System.out.println("JDBC Driver not found");  
    e.printStackTrace();  
}
```


៦. ស្វ័យប្រវត្តិ Connection (ត)

❖ Old Way (Explicit Driver Registration) (ត)

```
} catch (SQLException e) {  
    System.out.println("Connection failed");  
    e.printStackTrace();  
}  
finally {  
    try {  
        // Close the connection to release resources  
        if (connection != null) {  
            connection.close();  
            System.out.println("Connection closed");  
        }  
    }  
}
```

៦. ស្វ័យប្រវត្តិ Connection (ត)

❖ Old Way (Explicit Driver Registration) (តិចបំប៉ន)

```
    } catch (SQLException e) {  
        System.out.println("Failed to close connection");  
        e.printStackTrace();  
    }  
}  
}
```

៦. ស្វ័យប្រវត្តិ Connection (ត)

❖ New Way (Automatic Driver Loading)

```
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.SQLException;
public class NewConnectionExample {
    public static void main(String[] args) {
        // JDBC URL for MySQL database
        String url = "jdbc:mysql://localhost:3306/mydatabase";
        String username = "your_username";
        String password = "your_password";
        // Initialize connection object
        Connection connection = null;
```

៦. ស្វ័យប្រវត្តិ Connection (ត)

❖ New Way (Automatic Driver Loading) (ត)

```
try {  
    // Establish connection (DriverManager automatically loads the driver)  
    connection = DriverManager.getConnection(url, username, password);  
    System.out.println("Connected to the database!");  
  
    // Perform database operations here  
  
} catch (SQLException e) {  
    System.out.println("Connection failed");  
    e.printStackTrace();  
}
```

៦. ស្វ័យប្រវត្តិ Connection (តាមរយៈ)

❖ New Way (Automatic Driver Loading) (តាមរយៈ)

```
} finally {  
    try {  
        // Close the connection to release resources  
        if (connection != null) {  
            connection.close();  
            System.out.println("Connection closed");  
        }  
    } catch (SQLException e) {  
        System.out.println("Failed to close connection");  
        e.printStackTrace();  
    }  
}
```

៦. ឯកសារយោង

- <https://www.ibm.com/topics/data-modeling>
- [File and Database](#)
- <https://youtu.be/OGP2R29vzAw?si=8Z4h6QjO4gUililm>
- <https://youtu.be/OLmAZmBSwMo?si=3un1yrACjxEWOVY5>
- <https://www.geeksforgeeks.org/jdbc-drivers/>
- <https://www.w3schools.in/sql/database-concepts>
- <https://vertabelo.com/blog/database-modeling-techniques/>
- <https://www.tutorialspoint.com/jdbc/jdbc-db-connections.htm>



មជ្ឈមណ្ឌលកូរ៉េ សហវិវ អេច អ ឌី
Korea Software HRD Center



មូលនិធិ សហវិវ កូរ៉េ សម្រាប់ជំនួយជាសកល
Foundation for Korea Software Global Aid

សូមអរគុណ!