Database Systems: Lab 5

Extendible Hashing Practice

1. Suppose that we are using **extendible hashing** to manage an initially empty relation `Student`. Records with the following key values:

7, 9, 33, 18, 10, 32, 24, 38, 3, 29

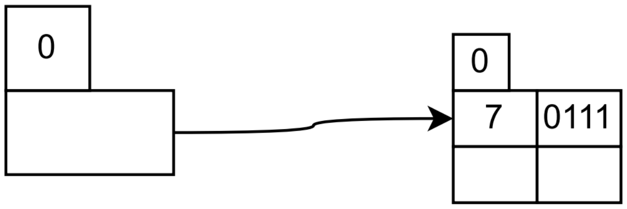
They are inserted to the relation sequentially. The **binary code** of each of the above keys can be found in the following table. Each block can hold up to **two records**.

Key value	**Hash code**
7	0111
9	1001
33	0001
18	0010
10	1010
32	0000
24	1000
38	0110
3	0011
29	1101

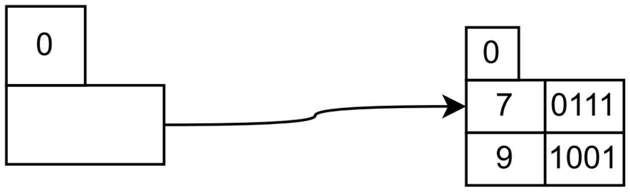
Show the **extendible hashing structure of whole insertion**, respectively. Please include the **global and local depths** in the answer.

Answer:

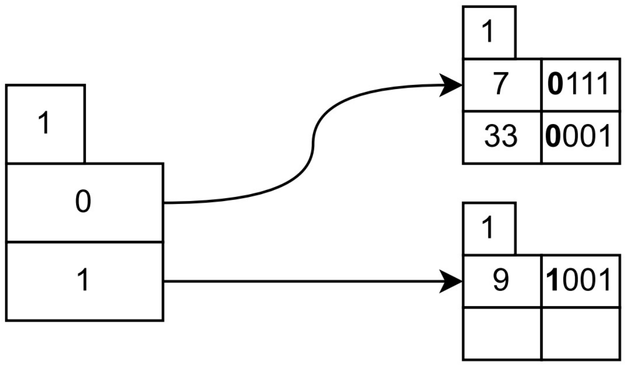
7:



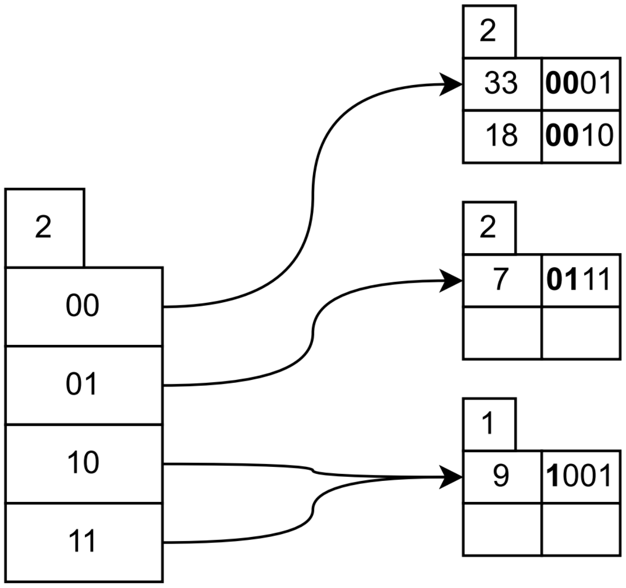
9:



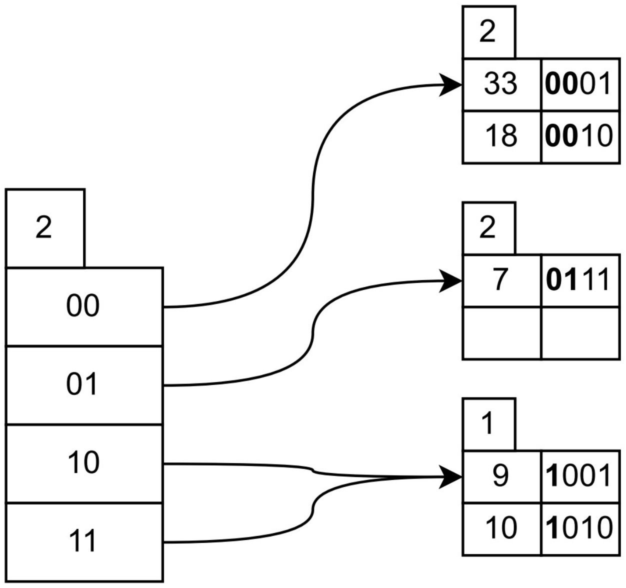
33:


1
0
1
1
7
0111
33
0001
1
9
1001

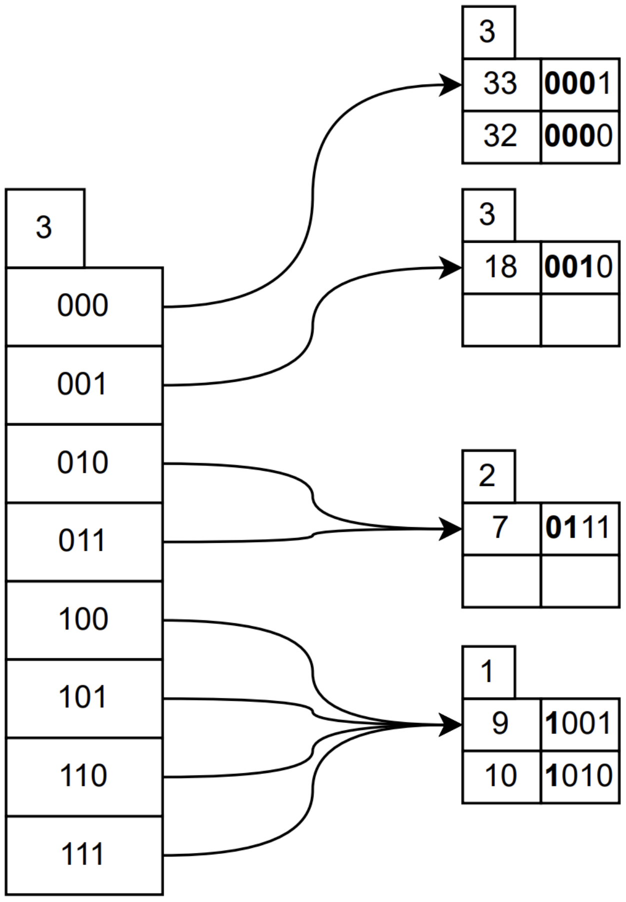
18:


2
00
01
10
11
2
33
0001
18
0010
2
7
0111
1
9
1001

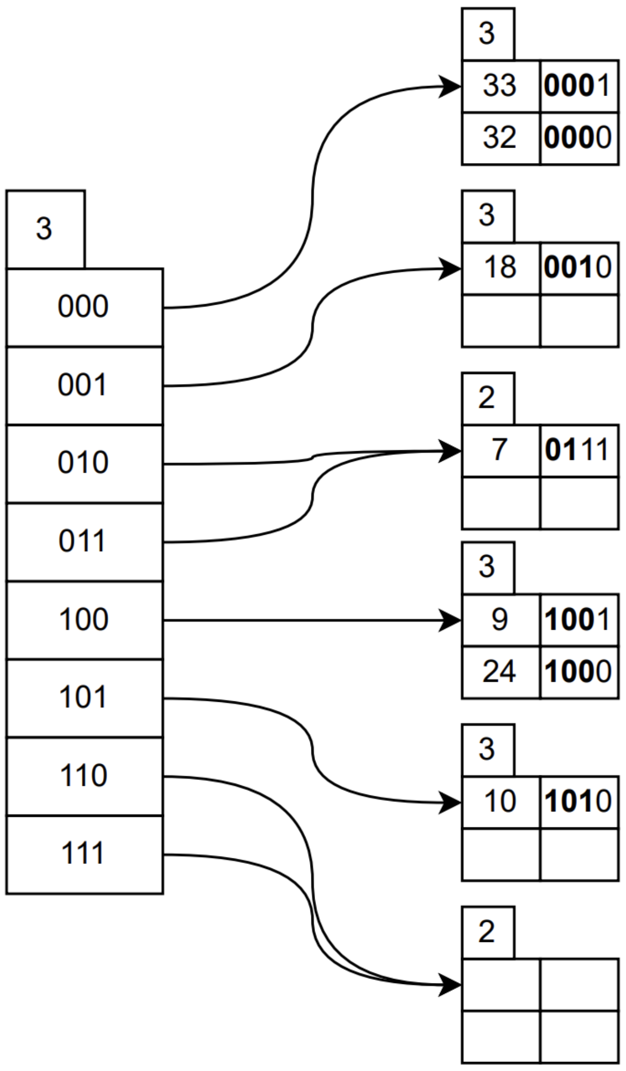
10:


2
00
01
10
11
2
33 0001
18 0010
2
7 0111
1
9 1001
10 1010

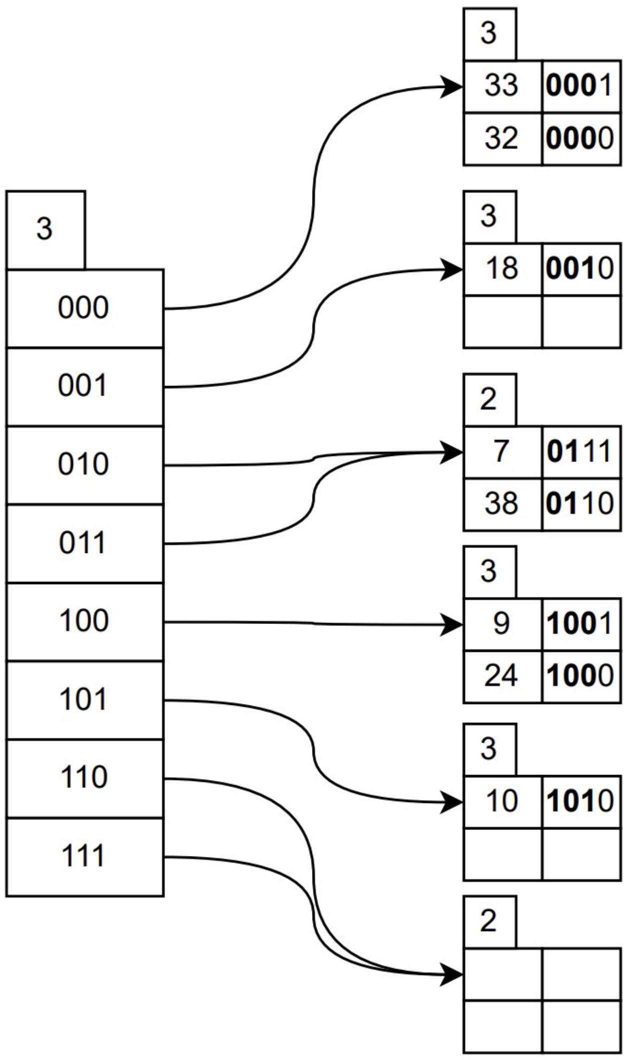
32:


3
000
001
010
011
100
101
110
111
3
33 0001
32 0000
3
18 0010
2
7 0111
1
9 1001
10 1010

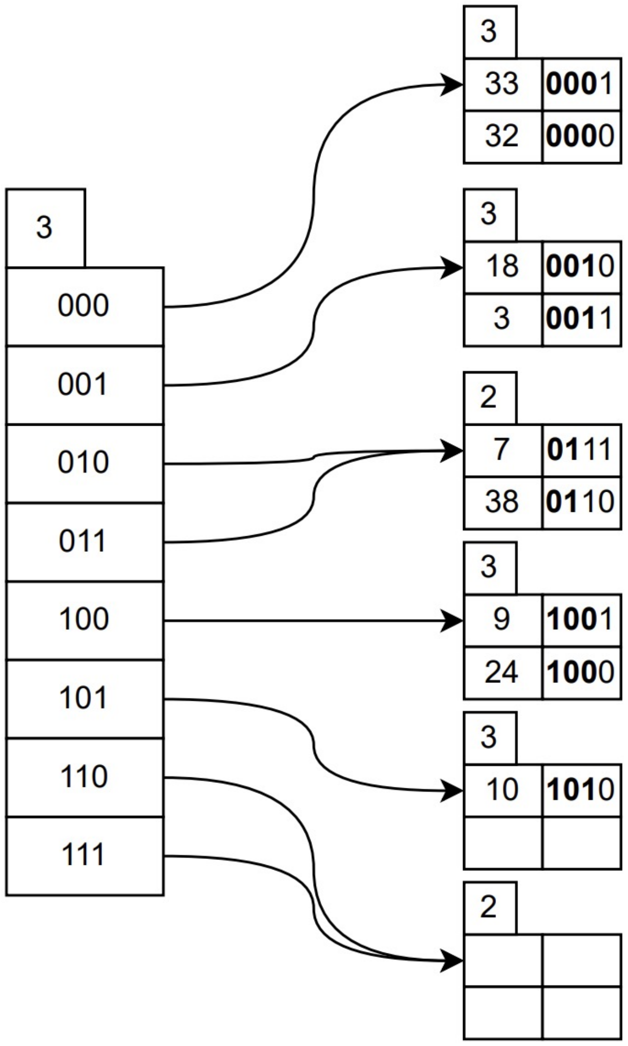
24:


3
000
001
010
011
100
101
110
111
3
33 0001
32 0000
3
18 0010
2
7 0111
3
9 1001
24 1000
3
10 1010
2

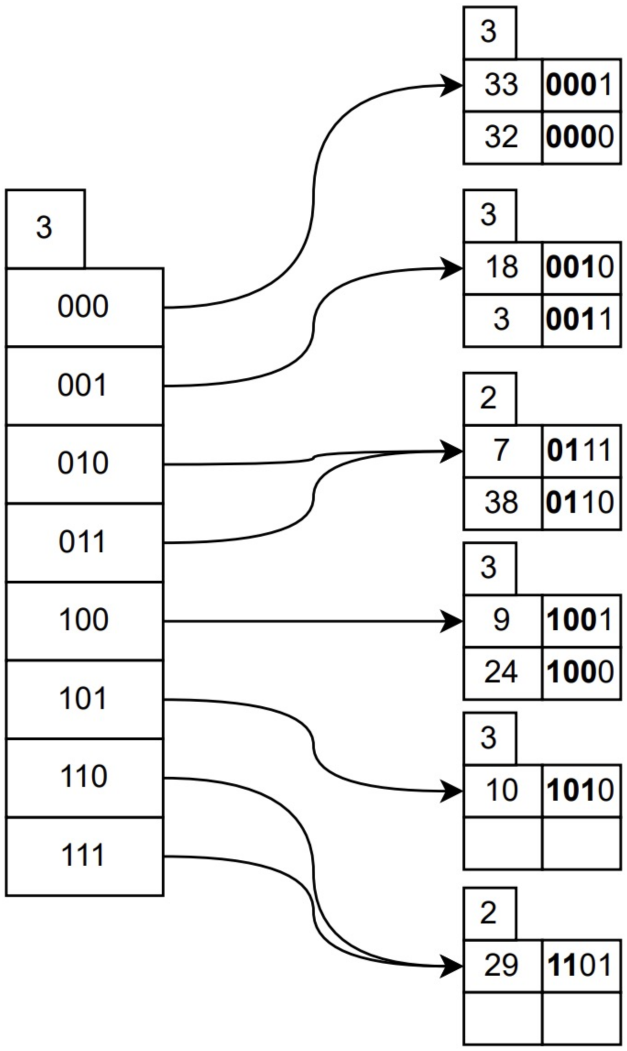
38:


3
000
001
010
011
100
101
110
111
3
33 0001
32 0000
3
18 0010
2
7 0111
38 0110
3
9 1001
24 1000
3
10 1010
2

3:


3
000
001
010
011
100
101
110
111
3
33 0001
32 0000
3
18 0010
3 0011
2
7 0111
38 0110
3
9 1001
24 1000
3
10 1010
2

29:



Database Access via Programming Language Practice

2. In this lab, your task is divided into three stages:

Stage 1: Initial Setup via PhpMyAdmin (Required Before Coding)

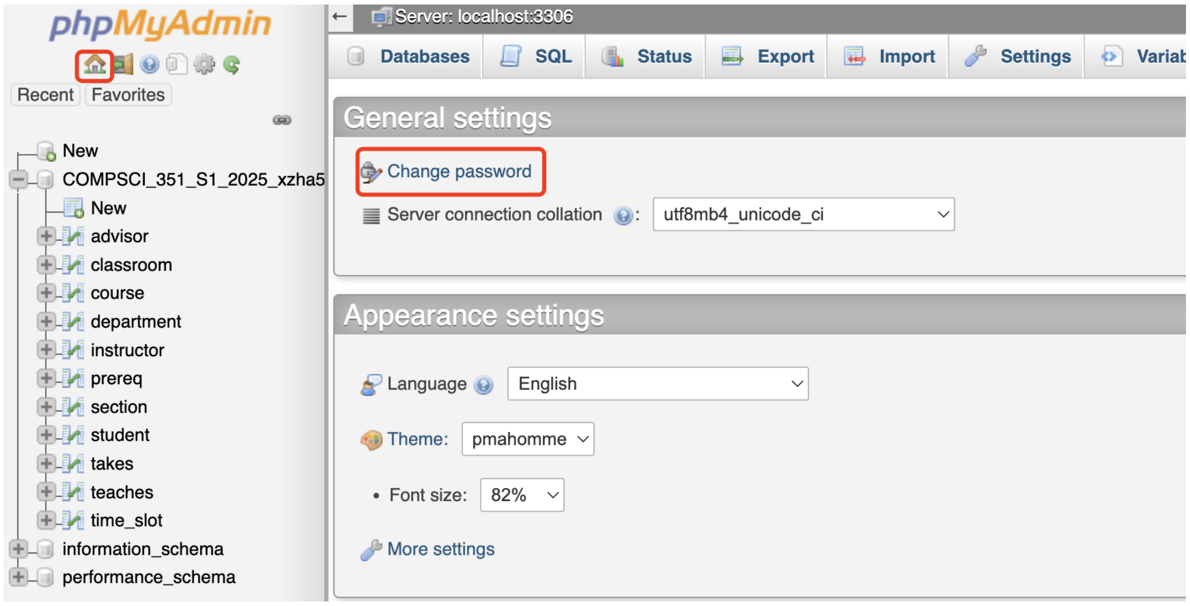
Before using the Programming Language program, you must first set up your Database environment via PhpMyAdmin:

1. **Check your university email inbox.** You should have received an email with the subject `"Database admin"`. If it is not in your inbox, please check your spam inbox first. If you still cannot find it, contact the tutors.

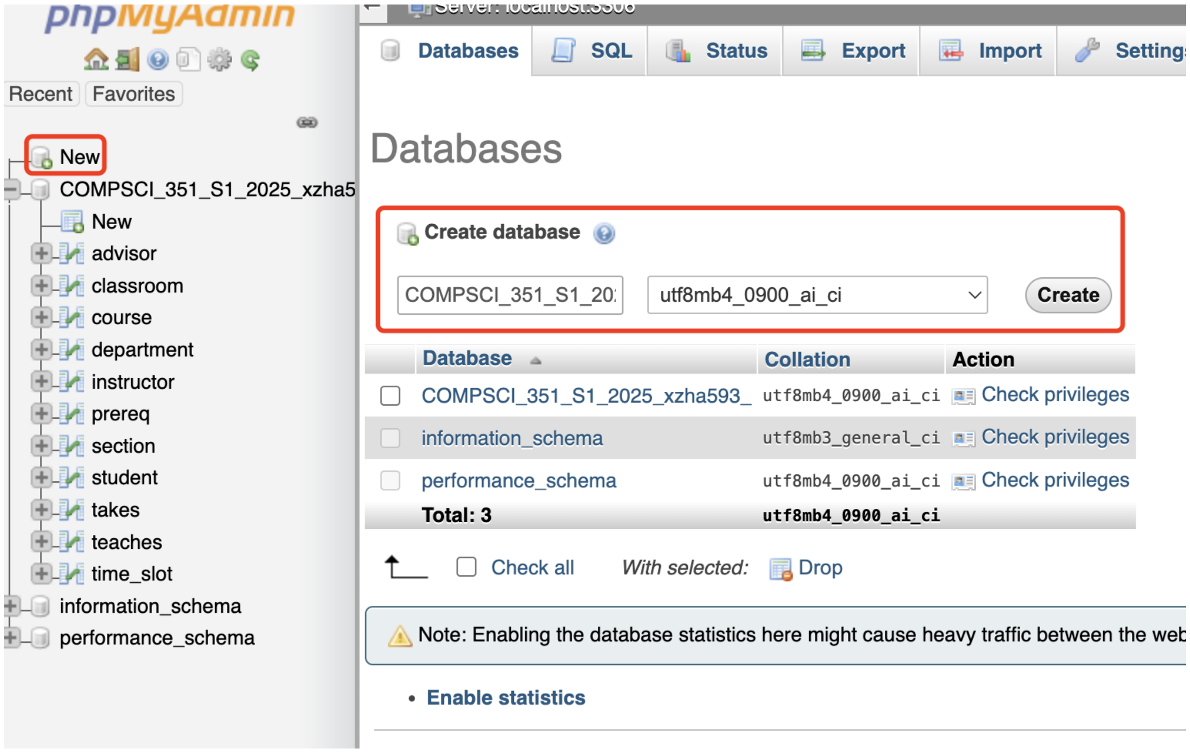
2. Visit: `fosmysqlprd01.its.auckland.ac.nz`. This is the host server for your database.

- **Important:** You can only access this link if you are connected to the campus network. If you are using your **personal device** outside the UOA network, please refer to `UOA VPN Setup (PDF)` for VPN setup instructions. For two-factor authentication (2FA), please refer to `Two-Factor Authentication Guide`. For detail please refer to **Stage 2.2**.
- **Recommended:** Use a lab computer at the university to complete this task.

3. **Change your MySQL password** after first login using **Native MySQL authentication** (not the default). This is done through the PhpMyAdmin interface.



4. **Create a new database**, then import and execute the SQL file `small_Relations.sql` to **create** the required tables and insert initial data.



Database	Collation	Action
COMPSCI_351_S1_2025_xzha593_	utf8mb4_0900_ai_ci	Check privileges
information_schema	utf8mb3_general_ci	Check privileges
performance_schema	utf8mb4_0900_ai_ci	Check privileges
Total: 3	utf8mb4_0900_ai_ci	

Note: Enabling the database statistics here might cause heavy traffic between the web

Stage 2: Set Up Your Python Environment

Choose one of the following options depending on whether you are using a university lab computer or your personal device:

1. **Using a University Lab Computer (LAN connection):**
   (a) After logging in to a lab computer, open a browser and download the "Portable Python" from: `Portable Python` (about 37MB).
   (b) Unzip the downloaded package into a folder of your choice, and run `Console-Launcher.exe` inside that folder.
   (c) In the command prompt that appears, install the MySQL connector by typing the following command (no admin rights required):
   ```
   pip install mysql-connector-python
   ```
   (d) Use `IDLE-Launcher.exe` in the same folder to launch Python and run the provided code: `Lab5.py`.
   (e) When prompted by the program, enter:
   - `Enter username:` — your UPI
   - `Enter password:` — your PhpMyAdmin password
2. **Using Your Own Device (Off-Campus or on UOA WiFi):**
   (a) For VPN setup instructions, please refer to **SECTION TWO** of the official VPN guide available at: `UOA VPN Setup (PDF)`.
   (b) If your FortiClient VPN login requires two-factor authentication (2FA), please refer to the official guide provided by the UOA available at: `Two-Factor Authentication Guide`.)
   (c) Once you have successfully connected to the FortiClient VPN, open your preferred Python IDE and run the provided code: `Lab5.py`.

Stage 3: Complete the Python Programming Tasks

MySQL Connector for Python provides a database connectivity mechanism for communicating with MySQL servers through a Python program.

The following is a simplified code segment that enables the login to your PhpMyAdmin server and displays the names of all databases.

Listing 1: Database access from Python

```python
import mysql.connector
from getpass import getpass
from mysql.connector import connect, Error

# Function: execute and print results of SQL query
def dbexec(conn, query):
    with conn.cursor() as cursor:
        cursor.execute(query)
        for tb in cursor:
            print(tb)

def main():
    try:
        with connect(
            host="fosmysqlprd01.its.auckland.ac.nz",
            user=input("Enter username: "),
            password=getpass("Enter password: "),
```

```
            autocommit=True
        ) as connection:

            # 1: Show all databases
            # TODO: Call dbexec with a query to show all databases
            pass

            # 2: Select a database and show its tables
            dbname = input("\nEnter a database: ")
            # TODO: Use the selected database
            # TODO: Show all tables in that database
            pass

            table = input("\nEnter a table name to SELECT * from: ")
            # TODO: Select and display all rows in the chosen table
            pass

            # 3: Insert a new instructor record
            print("Inserting a new instructor record into the table...")
            # TODO: Add insert statement

            try:
             # TODO: Call dbexec with a query to insert
                print("Insert successful.")
            except Error as e:
                print("Error:", e)

            # 4: Optional query loop
            while True:
                user_query = input("\n(Optional) Enter a SQL to execute or 'exit'
                    to finish: ")
                if user_query.lower() == "exit":
                    break
                try:
                    # TODO: Execute the user's query
                    pass
                except Error as e:
                    print("Error:", e)

    except Error as e:
        print(e)


main()
```

Once you're ready, run the simplified Python script `Lab5.py` we provide, or copy paste from before. You will then complete the following programming tasks:

1. **Select the database you already created on PhpMyAdmin** and display its tables. Use: `USE database_name` and `SHOW TABLES`
2. **Select the instructor table** and display all its rows. Use: `SELECT * FROM instructor`
3. **Insert a new row into the instructor table**. Use:

   ```
   INSERT INTO instructor (ID, name, dept_name, salary)
   VALUES (19990, 'TestUser', 'Finance', 70000.00)
   ```

   You will see the information you added in PhpMyAdmin.

4. **Optional:** Enter and execute any valid DML SQL query, and display the results. Continue accepting input until the user types `exit`.

A sample output of the completed program is shown as follows. Note that `"abc123"` is used as the example login UPI. You should replace it with your own student UPI value.

```
Enter username: abc123
Enter password: ************

('COMPSCI_351_S1_2025_abc123_',)
('information_schema',)
('performance_schema',)

Enter a database: (COMPSCI_351_S1_2025_abc123_) -< your input
('advisor',)
('classroom',)
('course',)
('department',)
('instructor',)
('prereq',)
('section',)
('student',)
('takes',)
('teaches',)
('time_slot',)

Enter a table name to SELECT * from: (instructor) -< your input
('10101', 'Srinivasan', 'Comp. Sci.', Decimal('65000.00'))
('12121', 'Wu', 'Finance', Decimal('90000.00'))
('15151', 'Mozart', 'Music', Decimal('40000.00'))
('22222', 'Einstein', 'Physics', Decimal('95000.00'))
('32343', 'El Said', 'History', Decimal('60000.00'))
('33456', 'Gold', 'Physics', Decimal('87000.00'))
('45565', 'Katz', 'Comp. Sci.', Decimal('75000.00'))
('58583', 'Califieri', 'History', Decimal('62000.00'))
('76543', 'Singh', 'Finance', Decimal('80000.00'))
('76766', 'Crick', 'Biology', Decimal('72000.00'))
('83821', 'Brandt', 'Comp. Sci.', Decimal('92000.00'))
('98345', 'Kim', 'Elec. Eng.', Decimal('80000.00'))
Inserting a new instructor record into the table...
Insert successful.

(Optional) Enter a SQL to execute or 'exit' to finish: (SELECT * FROM instructor) -< your input
('10101', 'Srinivasan', 'Comp. Sci.', Decimal('65000.00'))
('12121', 'Wu', 'Finance', Decimal('90000.00'))
('15151', 'Mozart', 'Music', Decimal('40000.00'))
('19990', 'TestUser', 'Finance', Decimal('70000.00'))
('22222', 'Einstein', 'Physics', Decimal('95000.00'))
('32343', 'El Said', 'History', Decimal('60000.00'))
('33456', 'Gold', 'Physics', Decimal('87000.00'))
('45565', 'Katz', 'Comp. Sci.', Decimal('75000.00'))
('58583', 'Califieri', 'History', Decimal('62000.00'))
('76543', 'Singh', 'Finance', Decimal('80000.00'))
('76766', 'Crick', 'Biology', Decimal('72000.00'))
```

```
('83821', 'Brandt', 'Comp. Sci.', Decimal('92000.00'))
('98345', 'Kim', 'Elec. Eng.', Decimal('80000.00'))

(Optional) Enter a SQL to execute or 'exit' to finish: (exit) -< your input
```

Answer:

```python
import mysql.connector
from getpass import getpass
from mysql.connector import connect, Error

def dbexec(conn, query):
    with conn.cursor() as cursor:
        cursor.execute(query)
        for tb in cursor:
            print(tb)

def main():
    try:
        with connect(
            host="fosmysqlprd01.its.auckland.ac.nz",
            user=input("Enter username: "),
            password=getpass("Enter password: "),
            autocommit=True
        ) as connection:
            # 1: Show all databases
            dbexec(connection, "SHOW DATABASES")

            # 2: Select a database and show its tables
            dbname = input("\nEnter a database: ")
            dbexec(connection, f"USE {dbname}")
            dbexec(connection, "SHOW TABLES")

            table = input("\nEnter a table name to SELECT * from: ")
            dbexec(connection, f"SELECT * FROM {table}")

            # 3: Insert a new instructor record
            print("Inserting a new instructor record into the table...")
            insert = """
            INSERT INTO instructor (ID, name, dept_name, salary)
            VALUES (19990, 'TestUser', 'Finance', 70000.00)
            """
            try:
                dbexec(connection, insert)
                print("Insert successful.")
            except Error as e:
                print("Error:", e)

            # 4: Optional query loop
            while True:
                user_query = input("\n(Optional) Enter a SQL to execute or 'exit'
                    to finish: ")
                if user_query.lower() == "exit":
                    break
                try:
                    dbexec(connection, user_query)
                except Error as e:
                    print("Error:", e)
    except Error as e:
        print(e)

main()
```