Q5

(a)

By prefixing the program variable by a colon(:) in SQL statement

EXEC SQL

Prefix

Preprocessor separates embedded SQL statements from host language code

Terminated by a matching END-EXEC

Or by a semicolon (;)

Shared variables

Used in both the C program and the embedded SQL statements

Prefixed by a colon (:) in SQL statement

Connecting to the database

CONNECT TO <server name>AS <connection name>

AUTHORIZATION <user account name and password> ;

Change connection

SET CONNECTION <connection name> ;

Terminate connection

DISCONNECT <connection name> ;

SQLCODE and SQLSTATE communication variables

Used by DBMS to communicate exception or error conditions

SQLCODE variable

0 = statement executed successfully

100 = no more data available in query result

< 0 = indicates some error has occurred

SQLSTATE

String of five characters

'00000' = no error or exception

Other values indicate various errors or exceptions

For example, '02000' indicates 'no more data' when using SQLSTATE

EX:

```
//Program Segment E1:
int loop;
EXEC SQL BEGIN DECLARE SECTION;
varchar dname [16], fname [16], lname [16], address [31];
```

```
char ssn [10], bdate [11], sex [2], minit [2];
float salary, raise;
int dno, dnumber;
int SQLCODE;
char SQLSTATE [6];
EXEC SQL END DECLARE SECTION;
loop = 1;
while (loop)
{
        prompt("Enter a Social Security Number: ", ssn);
        EXEC SQL
                SELECT Fname, Minit, Lname, Address, Salary
                INTO :fname, :minit, :lname, :address, : salary
                FROM EMPLOYEE WHERE Ssn = : ssn;
        if (SQLCODE == 0)
                 printf(fname, minit, lname, address, salary);
        else
                 printf("Social Security Number does not exist: ", ssn);
        prompt("More Social Security Numbers (enter 1 for Yes, o for NO):", loop);
}
```

(b)

Loop over the tuples in a query result

Points to a single tuple (row) from result of query

OPEN CURSOR command

Fetches query result and sets cursor to a position before first row in result

Becomes current row for cursor

可搭配 FETCH commands 使用

FETCH commands

Moves cursor to next row in result of query

Cursor point to a position before the first tuple (row) from result of query, and fetch command will let cursor point to the next tuple

(c)

Auto-global predefined PHP variable

Array that holds all the values entered through form parameters

Provides input values submitted by the user through HTML forms specified in <INPUT> tag

this contains an associative array of variables passed to the current script using a form submitted using the “POST” method

is an associative array containing data from a POST request.
is used to collect values from a form with method “post”

Q6

(a)

Common for Web applications

Clients

Provide user intervace. Let users can manupulate on it

Receiving user’s information and send to Intermediate Layer or presenting information from Intermediate Layer to users

Provide appropriate interfaces through a client software module to access and utilize the various server resources.

Clients may be diskless machines or PCs or Workstations with disks with only the client software installed.

Connected to the servers via some form of a network.

(LAN: local area network, wireless network, etc.)

Represents Web browser, a Java or other application, Applet, WAP phone etc. The client tier makes requests to the Web server who will be serving the request by either returning static content if it is present in the Web server or forwards the request to either Servlet or JSP in the application server for either static or dynamic content.

Intermediate Layer called Application Server or Web Server:

Provide appplication or logic operation

Stores the web connectivity software and the business logic part of the application used to access the corresponding data from the database server

Acts like a conduit for sending partially processed data between the database server and the client.

This layer provides the business services. This tier contains the business logic and the business data. All the business logic like validation of data, calculations, data insertion etc. Are centralized into this tier as opposed to 2-tier systems where the business logic is scattered between the front end and the backend. The benefit of having a centralized business tier is that same business logic can support different types of clients like browser, WAP (Wireless Application Protocol) client, other standalone applications written in Java, C++, C# etc. This acts as an interface between Client layer and Data Access Layer. This layer is also called the intermediary layer helps to make communication faster between client and data layer

Server

Provide access to database

Provides database query and transaction services to the clients

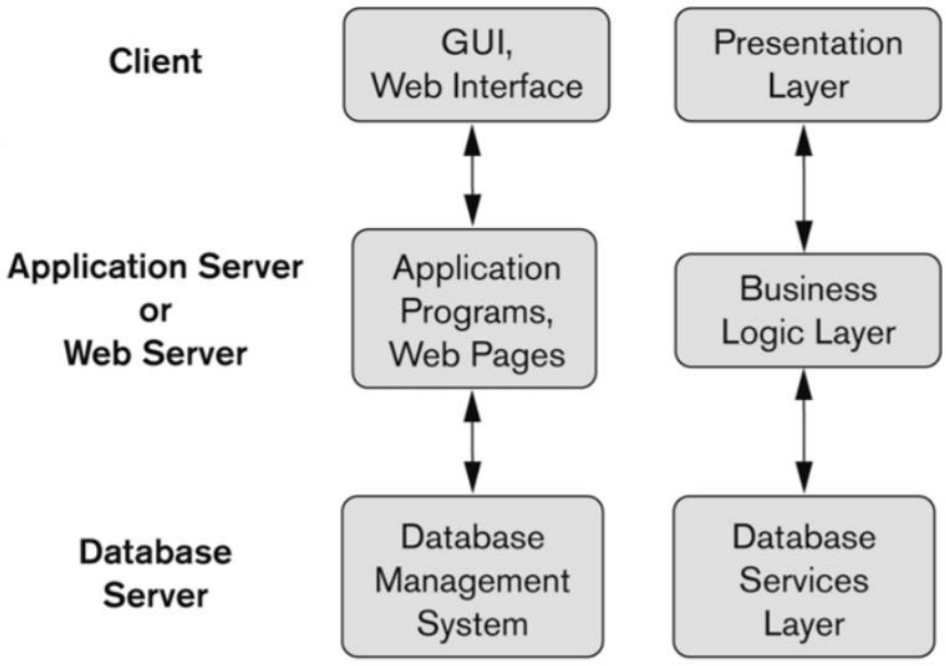
Relational DBMS servers are often called SQL servers, query servers, or transaction servers

Applications running on clients utilize an Application Program Interface (API) to access server databases via standard interface such as:

ODBC: Open Database Connectivity standard

JDBC: for Java programming access

This layer is the external resource such as a database, ERP system, Mainframe system etc. responsible for storing the data. This tier is also known as Data Tier. Data Access Layer contains methods to connect with database or other data source and to perform insert, update, delete, get data from data source based on our input data



(b)

Three-tier Architecture Can Enhance Security:

Database server only accessible via middle tier

Clients cannot directly access database server

Clients contain user interfaces and Web browsers

The client is typically a PC or a mobile device connected to the Web

High performance, lightweight persistent objects.

Scalability – Each tier can scale horizontally.

Performance – Because the Presentation tier can cache requests, network utilization is minimized, and the load is reduced on the Application and Data tiers.

Better Re-usability.

Improve Data Integrity.

Improved Security – Client is not direct access to database.

Forced separation of user interface logic and business logic.

Business logic sits on small number of centralized machines (may be just one).

Easy to maintain, to manage, to scale, loosely coupled etc.

Q7

Precompiler read the application program first. → extracting(DML command) → and send them to the DML compiler, then send the rest of the program to the host language compiler. The object code for DML command will finally be linked to the rest of the program and form a executable transaction

EX：

```
varchar fname [16], lname [16], address [31];
char minit [2];
float salary;
EXEC SQL
        SELECT Fname, Minit, Lname, Address, Salary
        INTO :fname, :minit, :lname, :address, : salary
Count: salary
```

Q8

(a)

(1) Class.forName("oracle.jdbc.driver.OracleDriver")

(2) Connection conn = DriverManager.getConnection
("jdbc:oracle:oci8:" + dbacct+ "/" + passwrd) ;

(3) p.clearParameters( ) ;

(4) p.setInt(1, dno) ; p.setString(2, sex) ; p.setDouble(3, sal) ;

(5) Lname = r.getString(1) ; salary = r.getDouble(2) ;

(b)

conn.prepareStatement(stmt1)：

stmt1 是一個 string

compile 檢查 stmt1 的語法是否正確並確認是否為真正的 sql 的 statement

according string stmt1 and connect the object with conn to creates a preparedstatement object

Creates a PreparedStatement object for sending parameterized SQL statements to the database.

If the driver supports precompilation, the method prepareStatement will send the statement to the database for precompilation.

r.next()：

loop 中每次抓下一筆，沒資料就跳出

The method ResultSet.next moves the cursor to the next row. This method returns false if the cursor is positioned after the last row. This method repeatedly calls the ResultSet.next method with a while loop to iterate through all the data in the ResultSet.

The next() method of the ResultSet interface moves the pointer of the current (ResultSet) object to the next row, from the current position.

on calling the next() method for the first time the result set pointer/cursor will be moved to the 1st row (from default position).

And on calling the next() method for the second time the result set cursor will be moved to the 2nd row.

Q9

(a)

```
function project_manager($managing, $project)
{
    if (array_key_exists($project, $managing))
    {
        $manager = $managing[$project];
        RETURN "$manager is managing $project.\n";
    }
    else
    {
        RETURN "There is no $project project.\n";
    }
}
```

(b)

```
$managing = ['AI' => 'Kevin', 'IOT' => 'Smith', '5G' => 'Mary'];
```

(c)

```
$x = project_manager($managing, 'IOT');
print ($x);
$x = project_manager($managing, 'eCommerce');
print ($x);
```

Q10

(a)

(1) array($_POST['stu_major'], $_POST['stu_year'])

(b)

(2) $_POST['stu_major']

(3) $_POST['stu_year']

(c)

```
while ($r = $q->fetchRow())
{
        print " Student name: $r[0], Student grade: $r[1], Student address: $r[2] \n ";
}
```

(d)

```
foreach ($allresult as $r)
{
        print " course name: $r[0], credits: $r[1], description: $r[2] \n" ;
}
```