Three Tier Architecture

- Commonly used in large internet enterprises

**Internet**

**Webserver**

**Application Server**

**Database Server**

- Eg. Apache/Tomcat connects clients to database systems
- Eg. IBM Websphere Application Server, Jboss, SAP Netweaver, etc. Performs business logic like shopping cart, checkout etc
- Eg. IBM DB2, Oracle, MS SQL Server. Runs DBMS, performs queries and updates from app server
SQL Environment

- Schemas: tables, views, assertions, triggers
  - `CREATE SCHEMA <schema name>`
  - Your login id is your default schema
  - `SET SCHEMA <schema>`
  - A fully qualified table name is `<schema>..<table>`

- Catalogs: collection of schemas
  - Corresponds to “databases” in DB2

- Clusters: collection of catalogs
  - Corresponds to “database instance” in DB2
Client-Server Model

- CONNECT TO <server> AS <connection name> AUTHORIZATION
- DISCONNECT/CONNECT RESET/TERMINATE
- Session – SQL operations performed while a connection is active

- Programming API
  - Generic SQL Interface
  - Embedded SQL in a host language
  - True Modules. Eg. Stored procedures.

Can be on same machine or different machines
Two extremes of the integration spectrum:

• Highly integrated eg. Microsoft linq
  – Compiler checking of database operations

• Loosely integrated eg. ODBC & JDBC
  – Provides a way to call SQL from host language
  – Host language compiler doesn’t understand database operations.

• Requirements:
  – Perform DB operations from host language
  – DB operations need to access variables in host language
Networking Basics

Each network “card” has a unique MAC address.

- IP address assigned by network provider: static or DHCP
- Port number usually fixed by application type
- Eg. http URLs, DNS

---

Client Application

- Higher level protocols
- Port number
- IP address
- MAC address

DBMS Server

- Higher level protocols
- Port number
- IP address
- MAC address

DBMS servers use their own protocols (eg. DRDA)

Servers use a port that is known by its clients

Servers use static IP address + DNS name

Eg. http URLs, DNS

Internet
Remote Client Access

• Applications run on a machine that is separate from the DB server
• DBMS “thin” client
  – Libraries to link your app to
  – App needs to know how to talk to DBMS server via network
• DBMS “full” client layer
  – Need to pre-configure the thick client layer to talk to DBMS server
  – Your app talks to a DBMS client layer as if it is talking to the server

What information is needed for 2 machines to talk over a network?
Configuring DBMS Client Layer

• Tell the client where to find the server
  `db2 CATALOG TCPIP NODE mydbsrv REMOTE 123.3.4.12 SERVER 50001`

• Tell the client where to find the server
  `db2 CATALOG DATABASE bookdb AS mybookdb AT NODE mydbsrv`
Embedded SQL in C Programs

- DBMS-specific Preprocessor translates special macros to DB-specific function calls
- Pre-processor needs access to DBMS instance for validation.
- Executable needs to be bound to a specific database in a DBMS in order to execute
Connecting SQL & Host Language

• Need a way for host language to get data from SQL environment
• Need a way to pass values from host language to SQL environment
• Shared variables
  – DECLARE SECTION
  – In SQL, refer using :Salary, :EmployeeNo

```sql
EXEC SQL BEGIN DECLARE SECTION;
char EmployeeNo[7];
char LastName[16];
double Salary;
short SalaryNI;
EXEC SQL END DECLARE SECTION;
```
An Example of Embedded SQL C Program

#include <stdio.h>
#include <string.h>
#include <sql.h>
int main()
{
    // Include The SQLCA Data Structure Variable
    EXEC SQL INCLUDE SQLCA;

    // Define The SQL Host Variables Needed
    EXEC SQL BEGIN DECLARE SECTION;
    char EmployeeNo[7];
    char LastName[16];
    double Salary;
    short SalaryNI;
    EXEC SQL END DECLARE SECTION;

    // Connect To The Appropriate Database
    EXEC SQL CONNECT TO SAMPLE USER
db2admin USING ibmdb2;

    // Declare A Static Cursor
    EXEC SQL DECLARE C1 CURSOR FOR
SELECT EMPNO, LASTNAME, DOUBLE(SALARY)
FROM EMPLOYEE
WHERE JOB = 'DESIGNER';

    // Open The Cursor
    EXEC SQL OPEN C1;
An Example of Embedded SQL C Program

// If The Cursor Was Opened Successfully, while (sqlca.sqlcode == SQL_RC_OK)
{
    EXEC SQL FETCH C1 INTO :EmployeeNo,
        :LastName, :Salary, :SalaryNI;

    // Display The Record Retrieved
    if (sqlca.sqlcode == SQL_RC_OK)
    {
        printf("%-8s %-16s ", EmployeeNo,
            LastName);
        if (SalaryNI >= 0)
            printf("%lf\n", Salary);
        else
            printf("Unknown\n");
    }
}

// Close The Open Cursor
EXEC SQL CLOSE C1;
// Commit The Transaction
EXEC SQL COMMIT;
// Terminate The Database Connection
EXEC SQL DISCONNECT CURRENT;
// Return Control To The Operating System
return(0);

• A cursor is an iterator for looping through a relation instance.
• Why is a cursor construct necessary?
Updates

- SQL syntax except `where` clause require current of `<cursor>

```sql
EXEC SQL BEGIN DECLARE SECTION;
int certNo, worth;
char execName[31],
execName[31],
execAddr [256],
SQLSTATE [6];
EXEC SQL END DECLARE SECTION;
EXEC SQL DECLARE execCursor CURSOR FOR MovieExec;
EXEC SQL OPEN execCursor
while (1) {
    EXEC SQL FETCH FROM execCursor INTO :
execName, :execAddr, :certNo, :worth;
    if (NO_MORE_ TUPLES) break;
    if ( worth < 1000)
        EXEC SQL DELETE FROM MovieExec
            WHERE CURRENT OF execCursor;
    else
        EXEC SQL UPDATE MovieExec
            SET netWorth=2*netWorth
            WHERE CURRENT OF execCursor;
}
EXEC SQL CLOSE execCursor
```
Static vs Dynamic SQL

• Static SQL refers to SQL queries that are completely specified at compile time. Eg.

```sql
// Declare A Static Cursor
EXEC SQL DECLARE C1 CURSOR FOR
SELECT EMPNO, LASTNAME, DOUBLE(SALARY)
FROM EMPLOYEE
WHERE JOB = 'DESIGNER';
```

• Dynamic SQL refers to SQL queries that are not completely specified at compile time. Eg.

```sql
strcpy(SQLStmt, "SELECT * FROM EMPLOYEE WHERE JOB=");
strcat(SQLStmt, argv[1]);
EXEC SQL PREPARE SQL_STMT FROM :SQLStmt;
EXEC SQL EXECUTE SQL_STMT;
```