Roll No : | | | | | | | |

KLNCIT	CLASS TEST ANSWER KEY	Format No.:ACD11A-I
		Issue No. :01
		Rev No. :00

K.L.N. COLLEGE OF INFORMATION TECHNOLOGY POTTAPALAYAM – 630 612
B.E. / B.Tech DEGREE COURSES (REGULATION -2013)
Class Test I / July / 2017

IT6701 / Information Management

Branch: Information Technology **Year / Sem : IV / VII**

Date / Duration: 17.07.2017 / 50 minutes **Max: 25 Marks**

I. Course outcomes, Question Number, Marks

COs	CO1	CO2	CO3	CO4	CO5
Q.Nos	1,2,3,4,5,6(a)/6(b)				
Marks (Max)	25				

II. Knowledge skill outcomes

Level	Remember (K1)	Understand (K2)	Apply (K3)	Analysis (K4)	Evaluate (K5)	Create (K6)
Q.Nos	1,3	2,4	5	-	-	6 (a) /6 (b)
Marks (Max)	4	4	2	-	-	15/15

PART – A **5*2=10 Marks**

Answer all the questions

1. List the types of data models. **(K1)**

The conceptual model is also known as the data model as data model can be used to describe the conceptual schema when a database system is implemented. It hides the internal details of physical storage and targets on describing entities, datatype, relationships and constraints.

A logical data model or logical schema is a data model of a specific problem domain expressed independently of a particular database management product or storage technology (physical data model) but in terms of data structures such as relational tables and columns, object-oriented classes, or XML tags.

A physical data model (or database design) is a representation of a data design as implemented, or intended to be implemented, in a database management system. In the lifecycle of a project it typically derives from a logical data model, though it may be reverse-engineered from a given database implementation.

2. Contrast JDBC and ODBC. **(K2)**

Java Database Connectivity (JDBC) is an application programming interface (API) for the programming language Java, which defines how a client may access a database.

Open Database Connectivity (ODBC) is an open standard application programming interface (API) for accessing a database. By using ODBC statements in a program, you can access files in a number of different databases, including Access, dBase, DB2, Excel, and Text.

3. Name any two sources of Business rules. **(K1)**

1. Business rules, the foundation of data models, are derived from policies, procedures, events, functions,and other business objects, and they state constraints(limit) on the organization.
2. Business rules represent the language and fundamental structure of an organization.

4. Illustrate stored procedure with an example. **(K3)**

A stored procedure is an ordinary program that can be called by an application with an SQLCALL statement.The stored procedure can be called locally or remotely. A remote stored procedure provides the most advantages:It reduces traffic across the communication line.It splits the application logic and encourages an even distribution of the computational workload.It provides an easy way to call a remote program.

```
CREATE PROCEDURE UPDATE_SALARY_1        (1)
        (IN EMPLOYEE_NUMBER CHAR(6),          (2)
        IN RATE INTEGER)                (2)
        LANGUAGE SQL                    (3)
        BEGIN
          UPDATE EMPLOYEE               (4)
          SET SALARY = SALARY * (1.0 * RATE / 100.0 )
          WHERE SSN = EMPLOYEE_NUMBER;
        END
```

5. Describe the connection object. **(K2)**

Connection Object The ADO Connection Object is used to create an open connection to a data source. Through this connection, you can access and manipulate a database.In JDBC, to obtain a connection, Initialize JDBC driver by calling the method Class.forName .This methods required an object of type java.sql.Driver . Each JDBC driver contains one or more classes that implements the interface java.sql.Driver .

PART – B **1*15=15 Marks**

6. (a) Develop a program to establish Java Database connectivity **(K6)**

```
//STEP 1. Import required packages

import java.sql.*;
public class FirstExample {
// JDBC driver name and database URL
static final String JDBC_DRIVER = "com.mysql.jdbc.Driver";
static final String DB_URL
= "jdbc:mysql://localhost/EMP";
// Database credentials
static final String USER = "username";
static final String PASS = "password";
public static void main(String[] args) {
Connection conn = null;
Statement stmt = null;
try{
//STEP 2: Register JDBC driver
Class.forName("com.mysql.jdbc.Driver");
//STEP 3: Open a connection
System.out.println("Connecting to database...");
conn = DriverManager.getConnection(DB_URL,USER,PASS);
//STEP 4: Execute
a query
System.out.println("Creating statement...");
stmt = conn.createStatement();
String sql;
sql = "SELECT id, first, last, age FROM Employees";
ResultSet rs = stmt.executeQuery(sql);
//STEP 5: Extract data from resu
lt set
while(rs.next()){
//Retrieve by column name
int id = rs.getInt("id");
```

```
int age = rs.getInt("age");
String first = rs.getString("first");
String last = rs.getString("last");
//Display value
s
System.out.print("ID: " + id);
System.out.print(", Age: " + age);
System.out.print(", First: " + first);
System.out.println(", Last: " + last);
}
//STEP 6: Clean
-
up environment
rs.close();
stmt.
close();
conn.close();
}catch(SQLException se){
//Handle errors for JDBC
se.printStackTrace();
}catch(Exception e){
//Handle errors for Class.forName
e.printStackTrace();
}finally{
//finally block used to close
resources
try{
if(stmt!=null)
stmt.close();
}catch(SQLException se2){
}// nothing we can do
try{
if(conn!=null)
conn.close();
}catch(SQLException se){
se.printStackTrace();
}//end finally try
}//end try
System.out.println("Goodbye!");
}//end main
}//end FirstExample
```

(OR)

6. (b) Create a program to implement the MapReduce (K6)

```
package hadoop;
import java.util.*;
import java.io.IOException;
import java.io.IOException;
import org.apache.hadoop.fs.Path;
import
org.apache.hadoop.conf.*;
import org.apache.hadoop.io.*;
import org.apache.hadoop.mapred.*;
import org.apache.hadoop.util.*;
public class ProcessUnits
{
//Mapper class
public static class E_EMapper extends MapReduceBase implements
Mapper<LongWrit
able, /*Input key Type */
Text, /*Input value Type*/
Text, /*Output key Type*/
IntWritable> /*Output value Type*/
{
```

```
//Map function
public void map(LongWritable key, Text value,
OutputC
ollector<Text, IntWritable> output, Reporter reporter) throws
IOException
{
String line = value.toString();
String lasttoken = null;
StringTokenizer s = new StringTokenizer(line,"
\
t");
String year = s.nextToken();
while(s.hasMoreTokens()){
lasttoken=s.nextToken();
}
int avgprice = Integer.parseInt(lasttoken);
output.collect(new Text(year), new IntWritable(avgprice));
}
}
//Reducer class
public static class E_EReduce extends MapReduceBase implements
Reducer< Text, IntWritable, Text, IntWritable >
{
//Reduce function
public void reduce(Text key, Iterator <IntWritable> values,
OutputCollector>Text, I
ntWritable> output, Reporter reporter) throws
IOException
{
int maxavg=30;
int val=Integer.MIN_VALUE;
while (values.hasNext())
{
if((val=values.next().get())>maxavg)
{
output.
collect(key, new IntWritable(val));
}
}
}
}
//Main function
public static void main(String args[])throws Exception
{
JobConf conf = new JobConf(Eleunits.class);
conf.setJobName("max_eletricityunits
");
conf.setOutputKeyClass(Text.class);
conf.setOutputValueClass(IntWritable.class);
conf.setMapperClass(E_EMapper.class);
conf.setCombinerClass(E_EReduce.class);
conf.setReducerClass(E_EReduce.class);
conf.setI
nputFormat(TextInputFormat.class);
conf.setOutputFormat(TextOutputFormat.class);
FileInputFormat.setInputPaths(conf, new Path(args[0]));
FileOutputFormat.setOutputPath(conf, new Path(args[1]));
JobClient.runJob(conf);
}
}
```

Faculty Incharge **Course Coordinator** **HOD/IT**