**QUESTION 1**

The major advantage of Solution 3 is that it has a very low cost.

Answer: **True**

**QUESTION 2**

What is most important to the client is that their customers are satisfied with the solution.

Answer: **True**

**QUESTION 3**

There is an error in the matrix weights.

Answer: **False**

**QUESTION 4**

There is an error in the matrix totals.

Answer: **True**

**QUESTION 5**

If Solution 3 required only technical expertise that was already available in-house, then it would no longer be the “winner” of this analysis.

Answer: **True**

**QUESTION 6**

If cost were a much more important criterion, then Solution 3 might no longer be the “winner” of this analysis.

Answer: **False**

**QUESTION 7**

Solution 3 is not likely to require more technical expertise than is available.

Answer: **False**

**QUESTION 8**

With Solution 1, we are very sure we know exactly how much it’s going to cost.

Answer: **False**

**QUESTION 9**

Customers are probably going to like Solution 2 more than Solution 1.

Answer: **False**

**QUESTION 10**

We have more information about the costs of Solution 2 than about the costs of Solution 1.

Answer: **True**

**QUESTION 11**

What is the difference between a requirement and a constraint?

**Answer:**

A constraint refers to any restriction that occurs on the degree of freedom one has in providing a solution to a problem. Constraint on the other side refers to the effective global requirements like limited resources to be used in the development or a decision by a senior management which restricts the project development methodology. Basically requirements describe what we need to happen while constraints describe the exact real-world boundaries/limits that surrounds the phenomena we want or happen.

**Comparison:**

In top-down versus bottom-up relationship, requirements are top-down i.e. starting from strategy to the execution phase while constraints are bottom-up.

In inside-out versus outside-in comparison, constraints are outside-in while requirements are inside-out

In enterprise-architecture vs solution-architecture comparison, enterprise-architects is mainly focused on requirements while solution-architects mainly focuses more on constraints.

In risk vs opportunity relationship comparison, requirements are often viewed as much aligned to opportunity, constraints are often regarded primarily as risks.

Requirements are usually brought together ether through workshops with the stakeholders, RFP or tender documents. Every requirement can be validated, and at regular intervals throughout delivery of the project. If a particular requirement is not met at any point during the project lifecycle – knowing about it sooner rather than later is preferred.

When providing a solution to a project, requirements are breaking down to two different categories:

1. Functional requirements which detail what the stakeholders expect from the solution, and what the end result should look like.
2. Non-functional requirements are usually an extra type of requirement that details performance and availability metrics around the solution taking into consideration all the stakeholders views – these could also be known as constraints around requirements.

Constraint are usually technical limitation that surrounds the solution, which could result in additional functional requirements that need to be captured up front. But a constraint can also be a range of values of a component of the system which then becomes a non-functional requirement.

**QUESTION 12**

When modeling, what is the difference between a current logical and a proposed physical data flow diagram?

**Answer:**

A data flow diagram (DFD) can be sectioned into two categories i.e logical or physical. A logical DFD mainly focuses on business activities and the business itself, while a physical DFD mainly focuses on how a system is implemented. Thus, when any data flow diagram maps out the flow of information of a process/system/or a project, the logical diagram provides the answer for the question (**what?**) and the physical DFDs provides answer to the question **(how?).** The logical DFD describes the business events that take place and the data required for each event.  It provides a solid basis for the physical DFD, which depicts how the data system will work, such as the hardware, software, paper files and people involved.

In summary the Logical Vs Physical DFD Difference is as follows:

1. **Logical DFD**

* Logical DFD depicts how the business operates.
* It shows business controls.
* The processes represent the business activities.
* The data stores represent the collection of data regardless of how the data are stored.

1. **Physical DFD**

* Physical DFD depicts how the system will be implemented (or how the current existing system operates).
* The processes represent the programs, program modules, and manual procedures.
* The data stores represent the physical files and databases, manual files.
* It describes controls for validating input data, for ensuring successful completion of a process, for obtaining a record, and for system security.

Look the table below for more description:

|  |  |  |
| --- | --- | --- |
| **Design Feature** | **Logical** | **Physical** |
| What the model depicts | How the business operates. | How the system will be implemented or how the current system operates. |
| What the processes represent | Business activities. | Programs, program modules, and manual procedures. |
| What the data stores represent | Collections of data regardless of how the data are stored. | Physical files and databases, manual files. |
| Type of data stores | Show data stores representing permanent data collections. | Any processes that operate at two different times must be connected by a data store. |
| System controls | Show business controls. | Show controls for validating input data, for obtaining a record |

**QUESTION 13**

Ocean City Airlines regularly updates the database containing their flight schedules. To represent this activity in the DFD above, the analyst will add a data flow entitled “new schedule” between “airline” and “flight schedule.” Is this correct?

**Answer:** B. No, because you can’t have a data flow between an external entity and a data store.

**QUESTION 14**

How many data flows would the context level diagram for this ticketing system have?

**Answer**: D. 6

**QUESTION 15**

How many external entities would the context level diagram for this ticketing system have?

**Answer:** C. 3

**QUESTION 16**

One detail missing from the DFD is that copies of tickets that are generated are not stored anywhere. Assuming that the ticketing system actually does store information about the tickets that are generated, what would be the simplest way to represent this on the diagram?

**Answer:** D. Add a data store called “D2 Issued tickets”, and a data flow called “Issued ticket info” from process 4.0 to D2

**QUESTION 17**

Which of the following is a violation of the Law of Conservation of Data?

**Answer: C.** Data flow “Credit Verification”

**QUESTION 18**

Going back to the original diagram as pictured, suppose that the systems analysis team also wanted to present a very low-end solution that has much less functionality. In particular, they wanted to suggest a system that requires the passenger to enter a specific flight number and date to purchase a ticket, i.e. the system no longer searches for flights based on the customer’s preferences, but the customer instead has to specify just one flight to purchase. Which of the following would be removed from the diagram to represent this option?

**Answer: A.** Data flows “Available flights”, “Potential flights”, “Flight choices”, and “Flight preferences”.

**QUESTION 19**

To create a physical level 0 DFD for this system, you would:

**Answer: B.** Add words describing the physical implementation details to the names of processes, data stores, and data flows.

**QUESTION 20**

Check the data flows that should be on the level 0 data flow diagram:

Confirmation from 5.0 to Employer﻿

Employer Information from Employer to 1.0﻿﻿﻿﻿﻿

New Employer Record from 1.0﻿﻿﻿

Potential Employees from D3 to D1﻿﻿﻿

Employer Information from Employer to D2﻿﻿﻿﻿

Employment Request Information from Employer to 2.0﻿﻿﻿﻿﻿﻿

Approved Contract from Employee to Employer﻿﻿

Contract Details from 4.0 to D3﻿

New Employment Request from 2.0 to D2﻿

Contract from 4.0 to Employee﻿

Signed Contract from Employee to 5.0﻿﻿﻿﻿﻿

Temporary Employment Request from 2.0 to D3﻿﻿﻿

Updated Employee Information from Employee to D3﻿﻿﻿

Rejected Contract from Employee to 3.0

Employee Review from Employer to 4.0﻿﻿

**QUESTION 21**

If you were to create an ER diagram for the Technical Temporaries system, which of the following would NOT be one of the entity types on the diagram?

**Answer: D.** Requests

**QUESTION 22**

Data are represented in all of the following diagrams. Which of them shows the structure of the data in the most detail?

**Answer: A.** Entity relationship diagrams

**QUESTION 23**

The difference between an entity type and an entity instance is analogous to the difference between

**Answer: C.** A species and an animal

**QUESTION 24**

A context level data flow diagram

**Answer: A.** Shows the inputs and outputs of a system

**QUESTION 25**

The cardinality of a relationship on an ER diagram specifies

**Answer: C.** How many instances of one entity type may be related to each instance of the other entity type?