1. **Give an example when you had to present complex information in a simplified manner in order to explain it to others.**

Explaining inner, outer , full DB joins to a student by comparing numbers in 2 baskets.

There are 2 baskets with numbers ( Venn diagram goes here) :

**Left Right**

**Basket TbA**

1. **Basket TbB**

2 2

3 3

4

1. **Show only numbers which present in both : A and in B (overlap/ intersect)**

**on the same row next to each other**

2-2

3-3

1. **For each** **number in A (Left) show if there is a match in B , if there is no match - put X**

1-X

2-2

3-3

1. **For each** **number in B (Right) show if there is** **match in A , if there is no match - put X**

2-2

3-3

X-4

1. To show **ALL numbers from B (Right) and A(Left) together , if there is no match put X**

1-X

2-2

3-3

X-4

To do the same with SQL query use syntax:

**select \***

**from TbA A (type) join TbB B**

**on A.num = B.num**

**type** allows to specify how you match the numbers

1. Key words :  **only numbers which present in both** specify **INNER** join

select \* from TbA A INNER join TbB on A.num = B.num

2. Key words:  **For each** **number in A (Left)**  specify **LEFT** join

select \* from TbA A LEFT join TbB on A.num = B.num

3.Key words :  **For each** **number in B (Right)** specify **RIGHT** join

select \* from TbA A RIGHT join TbB B on A.num = B.num

4.Key words **ALL from TbA and TbB** specify **FULL(FULL OUTER)**

select \* from TbA A FULL OUTER join TbB B on A.num = B.num

1. **Provide an example of a situation where you had to make a decision that your team originally disagreed with. What did you learn?**

In one of the projects we just started , I was one of the of contractors without business knowledge and good understanding what the final product have to look like , so everybody , started pushing favorite technologies ,design patterns and proposed solutions - emails were flying back and forth.

Based on my limited understanding of the task, I wrote small module in C++ ( because I was working with it at that time) with simplified functionality which allowed to mimic the functionality we would need to implement and start discussion/see some results based on concrete example.

In the end final product didn’t look anything like my code.

The best way to convince somebody is a working prototype and you can always built on it, change it or come up with better prototype.

1. **Describe the most difficult team you’ve ever led. What was difficult and how did you cope?**

Being team lead /SME for offshore team. At one of the financial institutions I was working, I had been tasked with taking over a product (maintenance/ reverse engineering /enhancements) and given 4 developers from offshore team who would work locally and communicate/dealt with a larger group of developers working offshore.

I introduced processes including build/deployment steps and improved logging which allowed to have consistent results. After I passed all the details to offshore team and they took this part over I’ve been contacted by Support reporting unexplained results generated by the software.

As it turned out the offshore team didn’t follow the steps and would ignore build , deployments warnings even though the steps were clearly defined.

The most difficult was to deal with lack of transparency. The worst was when I would discover there were issues which as it turned out they knew about and worked around in a wrong way without clear understanding of what they were doing.

My attempts to communicate that it is OK to ask questions and make mistakes and be open about it failed.

Considering the offshore team had their own parallel management infrastructure in place, at some point I had to escalate the issues to my management which escalated it to offshore management team, after which we came up with me technical issues resolution processes when I could monitor technical issues and the details of how they were resolved.

1. **Tell me about a time when you had to work on something you’re not familiar with at all. How did you go about it? Were you successful?**

Each new project/product has some unfamiliar elements. To implement them you need to be successful.

I guess, research , information gathering and ability to stay open minded is the key.

Typically:

If you are working with existing product:

1.Getting top level overview of the product and technology stack( ideally some business details).

2.Looking at the existing code solution and reading available documents ( if any).

3.Getting access to Runtime environment where the product run

and looking at the logs (DB? if available) and produced results.

Bonus: if you can build, deploy and run the product yourself.

4.Talking to available sources who are familiar with the product:

Developers, Testers, BAs, PMs, Business stakeholders ( more information is better and each person can add another unique perspective) to confirm your findings/understanding.

5.If there are no official documentation come up with a document (plain English/simplified diagrams if needed) where you describe the product/functionality the way you captured it and sending document out/posting on Wiki asking to preview it people who you talked to.

6.When you achieved some level of familiarity with the product

check business requirements and think about how to go about the task.

Check whether you can reuse/adopt existing logic, if not : implement new logic , follow existing code patterns and technologies unless you have mandate to change it.

7.Prototype coding, when you have working prototype and ready to demonstrate it to people who you talked to, update document /posting on Wiki asking to preview it people who you talked to.

8.Repeat step 7 in iterations until prototype approved.

9.After prototype approved by stakeholders , follow the company development procedures to code/build/deploy and test the changes.

10.Document the changes : either update existing documents or add new ( step 5)

NOTE: you will not remember yourself what you did or why you did it in a month if you switch to another project

**If it is new product:**

1.Getting business requirements deciding on technology stack and dependencies , setting run time infrastructure.

2.Clarifying requirements enough to start coding prototype.

3.Prototype coding, when you have working prototype ready, demonstrate it

to stakeholders confirming that you are moving in a right direction .

4.Come up with a documentation (plain English/simplified diagrams if needed) where you describe the product/functionality the way you captured it and sending document out/posting on Wiki asking to preview it people who you talked to .

5.Repeat step 3-4 in iterations until prototype found satisfying business request criteria and approved.

6.After prototype approved by stakeholders , follow the company development procedures to code/build/deploy and test the product.

7.Document the changes