**1.  VERSIONS of Server.**  You have founded a new company with two friends. Your new application (app) uses a SQL Server database to store information. You are unsure whether your app will be successful but if it is, you will need both high performance and space for large volumes of data. However, you have not yet launched, so are unsure how many people will use your app. Which edition of SQL Server 2016 should you use for this system? Why?

- I would use the SQL Server 2016 Enterprise edition as it known for its high-end capabilities with fast performance, unlimited virtualization and end-to-end business intelligence.

**2.  What is predicate logic?** How is it a different way of thinking about data and the processing that you want to do on data? What does it mean?  What is a "predicate?"  Describe each of the below T-SQL elements as it relates to predicate:

- Predicate logic is another branch of mathematics on which the relational model is based.

- Using predicate logic into processing the data gives you more convenient ways to filter data from your database.

- Predicate is an expression that is either true or false.

* WHERE clauses – is used to filter records from the rows that fulfills its specified condition.
* JOIN conditions -  is used in the WHERE Clause of select, update, delete statements to join and combine rows from two or more tables.
* HAVING clauses – it is a group filter that only retrieves data that evaluates to TRUE from its specified condition.
* WHILE statements – it is used to execute code in a loop until the condition evaluates to FALSE.

**3. Going wild with wildcards.**  Using wildcards as arguments (such as Select \*) is not recommended. Why? What if you really need to do such a "wild" search, select, or filtering? What trade-offs should you consider? Alternatives?

- SELECT \* would not be appropriate if you have thousands of data on your database as it will show you ALL the unnecessary records on your database that you are not querying about. If you only want to filter certain elements, it best to use the FROM, WHERE and HAVING clauses to best help you query the data you only need.

**4.  JOINS.**  Inner and Outer joins are talked about a lot as we use tables and database systems.  What do these mean, and why you'd need to use them. What if you're operating in a non-SQL kind of environment, where you don't have JOIN operators? How might you accomplish the same results?

- INNER and OUTER joins are used to combine two or more tables into a single result. This conditions are used to limit which rows and columns are combined to match the query command. In a non-SQL environment, there are a lot of ways to manually join certain tables or information together such as using a venn diagram or by simply adding tables together.

**5. Procedural, Object, or Data: Perspectives?**  As a programmer, you have to focus your thinking either on the step-by-step of what you want to do, or the object-by-object, data-item-by-data-item that you need to do those steps to.  How does your viewpoint or thought process change as you "think in SQL?"

- Thinking in SQL will help me understand what SQL queries do, how to create and change them, and how to create reports from the data recovered. It is not a step-by-step process but rather a query-by-query process.

References:

<https://docs.microsoft.com/en-us/sql/sql-server/editions-and-components-of-sql-server-2016?view=sql-server-ver15>

<https://www.w3schools.com/sql/sql_where.asp>