Q: **What is SAP HANA?**

A: SAP HANA is an in-memory, column-oriented, relational database management system developed by SAP. It is designed to process large amounts of data in real-time, enabling businesses to make faster and more informed decisions.

Q: **What is SAP ABAP on HANA?**

A: SAP ABAP on HANA is a programming language used to develop applications in the SAP HANA database environment. ABAP on HANA provides enhanced performance and improved processing speed by leveraging the in-memory computing capabilities of the HANA database. It provides improved performance, real-time analytics, simplified development, and enhanced scalability.

Q.**What is difference between SAP HANA and S/4 HANA?**

SAP HANA is an in-memory database platform that enables real-time data processing and analysis. It can be used as a standalone database platform or as the underlying technology for other SAP applications.

SAP S/4HANA, on the other hand, is a suite of enterprise resource planning (ERP) applications that are built on top of the SAP HANA platform. S/4HANA is a next-generation business suite that is designed to help businesses run more efficiently by providing real-time insights and streamlined processes.

In summary, SAP HANA is a technology platform, while SAP S/4HANA is an ERP suite that is built on top of the SAP HANA platform.

Q: **What is the difference between traditional ABAP and ABAP on HANA?**

A: The main difference between traditional ABAP and ABAP on HANA is that ABAP on HANA uses the HANA database technology to process data faster and more efficiently. This means that ABAP on HANA applications can leverage the power of HANA to provide real-time data processing and analysis.

**What are the main features of SAP HANA?**

**In-memory Database:**

With SAP HANA, you can now store the complete database in memory. This means that disk movement is not needed and swapping can be eliminated. Hence increase in the speed of read-write access.

**Multicore CPU and parallel processing:**

To get the best performance from new advanced hardware, SAP HANA makes use of parallelism by using all the cores of a CPU, and several CPUs.

Column store tables are automatically processed in parallel.

Even the same column can be split up and processed by different cores at the same time.

**Column and Row storage:**

SAP HANA supports both col and row store tables

By accessing data in column-store order, you benefit immensely from simplified table scan and data pre-caching. This can make all the difference in performance.

With column store, SAP HANA scans columns of data so quickly that additional indexes are usually not required.

It is easy to alter column store tables without dropping and reloading data.

Column store tables are optimal for parallel processing, as each core is able to work on a different column.

**Data Compression**

SAP HANA uses a dictionary per column

Operates directly on compressed data using integers

It reduces the amount of memory required.

It speeds up operations on columns because the columns can be loaded into the CPU caches faster and with fewer loading cycles.

**Aggregates and Indexes.**

Using the power of SAP HANA, you can aggregate on the fly from any line item table. You do not need prebuilt aggregates.

SAP HANA organizes data using column stores, which means that indexes are not needed. They can still be created but offer little improvement.

**What is the in-memory database?**

Storing data in main memory rather than on disk provides faster data access, faster querying and processing.

SAP HANA uses Dynamic tiering, frequently accessed "hot" data is stored on main memory and less frequently accessed "warm" data is stored on disk.

Q: **What is the difference between a traditional database and SAP HANA?**

A: The main difference between a traditional database and SAP HANA is that SAP HANA is an in-memory database that stores data in columns rather than rows. This enables faster data processing and real-time analytics capabilities.

Q: **Advantages of column store over row store?**

A: you benefit immensely from simplified table scan and data pre-caching.

only the required columns are loaded to memory, so you avoid using up memory with columns that will never be used the data is arranged efficiently with all values of a column appearing one after another. This continuous sequencing of the column values is preferred by the CPU.

It is easy to alter column store tables without dropping and reloading data.

Q:**Guidelines to increase ABAP code performance on SAP HANA database?**

A:

* Keep result set small.
* Minimize the number of data transfer.
* Minimize number of database accesses.
* Minimize search overhead.
* Keep unnecessary load away from database.

Q: **Can we create secondary index in SAP HANA?**

A: Yes, A secondary index in SAP HANA is an additional index that can be created on a table to improve the performance of specific queries.

Q: **What are the benefits of data compression in SAP HANA?**

A: The benefits of data compression in SAP HANA include:

* Reduced memory and storage requirements
* Improved query performance due to reduced data volume
* Lower costs for hardware and storage
* Faster data backups and restores.

Q: **What are the benefits of parallel processing in SAP HANA?**

A: The benefits of parallel processing in SAP HANA include:

* Faster query performance due to the ability to process tasks concurrently.
* Improved scalability for larger workloads
* Ability to leverage distributed computing resources for improved processing power.
* Reduced hardware costs compared to traditional single-node architectures.

Q: **What are the benefits of SAP HANA multi-CPU architecture?**

A: SAP ABAP HANA multi-CPU architecture offers several benefits, including increased processing power, improved data processing speed, and high availability. By distributing the workload across multiple CPUs, the system can handle larger data volumes and execute complex queries faster. Additionally, the system can continue to operate even if one or more CPUs fail, ensuring high availability.

Q:**How is code push down implemented in SAP ABAP HANA?**

A: Code push down in SAP ABAP HANA is implemented through the use of new ABAP language constructs, such as inline declarations and expressions, which allow developers to perform complex database operations directly in the database layer. Additionally, SAP provides tools such as ABAP Managed Database Procedures (AMDPs) and Core Data Services (CDS) to help developers implement code push down techniques in their applications.

Q: **What are the benefits of code push down in SAP ABAP HANA?**

A: The benefits of code push down in SAP ABAP HANA include improved application performance, reduced network traffic and memory usage, and improved scalability. By pushing processing down to the database layer, the application can take advantage of the in-memory computing capabilities of the HANA database, resulting in faster data access and processing.

Q: **What is SAP ADT?**

A: SAP ADT stands for ABAP Development Tools. It is an integrated development environment (IDE) used by developers to develop, debug, and deploy ABAP applications in the SAP system.

Q: **Advantages of using ADT over ABAP workbench?**

A:

![A white sheet with black text

Description automatically generated]()

Q: **How do you create a new ABAP project in SAP ADT?**

A: To create a new ABAP project in SAP ADT, follow these steps:

1. Open the SAP Development perspective in Eclipse.
2. Right-click on the ABAP Development node in the Project Explorer and select New > ABAP Project.
3. Enter a name and description for the project, and select the system and client where the project will be created.
4. Select the package where the project will be saved and specify any additional project settings.
5. Click Finish to create the project.

Q: **How do you debug an ABAP program in SAP ADT?**

A: To debug an ABAP program in SAP ADT, follow these steps:

1. Open the ABAP Development perspective in Eclipse.
2. Navigate to the program you want to debug in the Object Navigator.
3. Set a breakpoint in the program by double-clicking on the line of code where you want to pause the program.
4. Execute the program in debug mode by right-clicking on the program and selecting Debug As > ABAP Application.
5. Step through the program line by line using the debug toolbar and inspect variables and objects in the Debug Perspective.