Q1. What is JVM, DVM & ART & list the difference between them?

**Java Virtual Machine**

A Java virtual machine (JVM) is a virtual machine that enables a computer to run Java programs as well as programs written in other languages that are also compiled to Java bytecode. The JVM is detailed by a specification that formally describes what is required in a JVM implementation. Having a specification ensures interoperability of Java programs across different implementations so that program authors using the Java Development Kit (JDK) need not worry about idiosyncrasies of the underlying hardware platform.

**Dalvik Virtual Machine**

The Dalvik Virtual Machine (DVM) is an android virtual machine optimized for mobile devices. It optimizes the virtual machine for memory, battery life and performance.

**Android Runtime** (**ART**)

Android Runtime (ART) is an application runtime environment used by the Android operating system. Replacing Dalvik, the process virtual machine originally used by Android, ART performs the translation of the application's bytecode into native instructions that are later executed by the device's runtime environment.

**List the difference between them?**

Conceptually, there is little difference from an application level between a DVM and a JVM. Architecturally, there is a major difference between the register based DVM and the stack-based JVM.

Both use a VM code model. However, the DVM uses register based opcodes that are comparable to the register-based bytecode instructions that most of the target platforms already execute. This includes architectures such as those available from ARM and MIPS and the x86-compatible architectures from Intel, AMD, and VIA Technologies.

Google developed Android and chose DVM for several reasons. First, there were licensing issues with most JVMs. Next, the DVM should be more efficient in terms of memory usage and performance on a register-based machine. DVM is also supposed to be more efficient when running multiple instances of the DVM. Applications are given their own instance. Hence, multiple active applications require multiple DVM instances. Like most Java implementations, the DVM has an automatic garbage collector.