Ans-1

(a) A Real time system is a system which is designed to handle events as they occur in real time. There is a great importance of speed at which tasks are completed.
These tasks are due to be performed within a specific time constraint.

Hard real time systems & soft real time Systems are both used in industry for different tasks.
The primary difference between hard and soft real time is that their consequences of missing a deadline differ from each other.

RTLinux has a special design for its kernal because it has two kernals. RTLinux uses a specialized real time kernal called RTcore.
The second kernal which is used for regular applications that donot have time constraints.
Both thread handling & interrupt handling are controlled by RTCore. This RTcore also restricts the Linux kernel by making it usable ~~to~~ disable interrupts to make sure it doesn't interfere with process scheduling.

Real time applications can communicate with Linux kernels through first in first out pipes.

Vxworks uses a single microkernel to handle basic kernel functions. Additional functions like file sharing and networking have to be loaded from provided functionality.

This system provides ~~[illegible]~~ flexibility to fit its functionality without loosening its constraints on available memory & resources.

As both of them are real time operating system and an autonomous robots is a complex system that has ability to decide its actions on its environment from its sensing, state in order to fill his aims. So we can prefer to use RTLinux / Vxworks for OS hardware and native robot software.

It is a very complex task but as both the RTOS contains schedulers which are preemptive in a nature and is flexible by allowing different scheduling techniques.

Vxworks contains kernal for preemptive multitasking, inter process communication and interrupt response.

So we can prefer both in hard / soft real time systems.

1 - b)

We will prefer Real time Databases for the autonomous robot because it is a database system which uses ~~[illegible]~~ real time processing to handle workloads whose state is constantly changing.

It uses ACID properties for timely execution of transactions. Here operations execute with predictable response, and with application acceptable levels of logical & temporal consistency of data, which is required in an autonomous robot implementation.

As the complexity of this task is very high, the amount of transactions to be handled by real time systems increases. So due to this we prefer real time databases.

Compared with traditional databases, real time database systems have a feature which must satisfy time constraints associated with transactions.
Transactions in real time DB systems should be scheduled considering both data consistency and timing constraints.
It must adapt to changes in the operating

environmat and guarantee the completion of critical transactions.
In this database system, the correctness of transaction processing depends not only on maintaining consitancy constraints and producing correct results but also on the time at which transaction is completed.
Transactions must be scheduled in such a way that they can be completed before their corresponding deadlines expire.

In traditional DB system, research has been focussed on providing high degree of predictability in task scheduling, the problem of integrating transaction processing capabilities with timing & consistency constraints has not received much attention.
Conventional dB systems are not used in real time applications due to two inadequancies: poor performance & lack of predictability.

Features of Real time Data Base system:

- Has timing constraints.
- queries to the DB should have soft & hard deadlines.
- Data returned must have both absolute consistency & relative consistency.

- Both real time scheduling & database technologies can be applied on real time data management

1-c) Due to the benefits of its reusability & productivity the component based approach has become the primary technology in robot s/w frameworks.

However all the existing systems are very limited in fault tolerance support, even though fault tolerance function is crucial for commercial success of service robots.

While substantial effort has been invested in making robots more reliable, experience demonstrates that robots operating in unstructured environments are often challenged by frequent failures.

Despite this, robots have not yet reached a level of design that allows effective management of faulty or unexpected behaviour by untrained users.

Information regarding failures that occurred for robot handling is classified in 4 categories:

→ Interaction
→ Algorithms
→ Hardware
→ Software.

They use several attributes to classify faults and

their properties including fault's tolerance to different robotic systems
→ tolerance
→ condition
→ symptoms

The influence of varying reliability on ~~t~~ real time system and found that periods of low reliability earlier during the interaction have a more negative impact on overall trust.

People trust a robot less when reliabity drops occurred late or in the middle of run.

So to handle all the faults and to increase reliability we can check all the possible test cases that might create problem while doing actual real time task.
Handling all these test cases will surely increase the reliability over the robots as a S~~ur~~urgeons. But there will be still chances of error as may possible that some new problem arise which was not handle before.