Context	Secure software development	
Component	Assessment	
Threat	Outdated infrastructure often forces internal development teams to use outdated libraries in their applications. This may be an impediment in the lifecycle management because updates may become difficult or impossible. Obsolete applications may have unpatched vulnerabilities.	
Vulnerability	Initial: blocking impediment in the lifecycle management of client applications Secondary: applications with dependencies not up-to-date may be exposed to unexpected cybersecurity risks	
Event	Initial: dependency scanner or pen test finds a vulnerability in an internal application. The development team can't upgrade the application because the obsolete dependency to access the database is incompatible with more recent releases of their frameworks. Secondary: cybersecurity incident happens. The development team cannot release an hotfix the vulnerability. Urgent solutions must be improvised to avoid further incidents.	
Asset at risk	Initial: none, but lifecycle management may become harder with time Secondary: data may be exposed, applications may be disrupted	
Likelihood rating	Initial: SQL technology is an industry-standard that is still actual. It is unlikely for the database to become completely unsupported in the next future resulting in a block of upgrades. It is however possible that specific SQL dialects or specific "oddities" of old databases may not be supported anymore. Rating 2 / 5	Subsequent: it is very likely for old dependencies to contain some kind of vulnerability. Not all vulnerabilities are always exploitable. Rating 4 / 5

Impact rating	Initial: not being able to perform normal lifecycle management has a generally mild impact in a short time and the impact grows over time as the application ages. It may eventually cause high costs when obsolescence requires complete rewriting. Rating 2 / 5	Subsequent: a security breach that cannot be quickly fixed due to a lack of maintenance is a major issue that poses a serious risk of business continuity. Rating 4 / 5
Impact target (who)	* development team, HR * company in general * customers	
Impact description (how)	* development team may be affected because hard lifecycle management and legacy software cause frustration. Recruitment may become a problem if the application becomes technologically obsolete * company reputation and economical damages may be a consequence of a security breach due to lack of upgrades * customers may be affected by a disruption in the service or by the leak of their data	
Risk rating	Initial: low 2 / 5	Subsequent: high 4 / 5
Potential control	Replacing all the database-specific functionalities with generic functions may reduce the risk of being locked to specific obsolete dependencies.	
Risk statement	It is recommended to ensure that internally developed applications follow industry standards to avoid vendor lock-in, especially when the infrastructure suffers of obsolescence	

Your recommendation to the risk owner	It is recommended to evaluate the costs to refactor the applications to enable the updates with generic modern dependencies. Business should evaluate the risk compared to the estimated costs. If costs are too high, it is recommended to plan a program to decommission the applications.

Likelihood rating	Meaning
1	Mostly a theoretical risk. Vulnerability cannot be exploited or event is very unlikely to happen
2	Improbable event. It is difficult to exploit the vulnerability, it is an uncommon technology, event rarely happens
3	Possible event. Given enough time the vulnerability will be exploited. Relatively common issue.
4	Probable event. It’s a well known vulnerability and there are tools to exploit it on the market. Event is very likely to happen during the year
5	Certain event. It’s a current active treat. Event may happen at any time

consequences rating	Meaning
1	Minimal or no consequences. Business is not disrupted.
2	Minimal disruption. Business may be affected within the terms of SLA

3	Moderate disruption. Violation of SLA. Potential financial or reputation consequences
4	Severe disruption. Financial or reputation consequences. Business impact can last for a short period
5	Catastrophic consequences. Potential permanent financial or reputation damages. Business may need long time to recover.

Question to ask	Expected response
What need to be protected?	Our database is very old and we have a couple of applications developed internally based on an ancient driver that works only on our old server. We are worried that we may have business continuity issues because maintenance is hard and upgrades are becoming impossible
Who will be using it and why?	The database is used by developers as a backend for the applications. Indirectly it is also used by some internal users and by customers.
How will it be stored?	N.A.
Where will it be stored?	Our applications are in a separate DMZ because we fear that they may become comprised.
Who/What are the threats and vulnerabilities	We have a dependency scanner and we perform pen tests. Sometimes we identify risks in our applications and we fix them. Unfortunately, now we have a minor vulnerability that requires a major upgrade, but the new version is incompatible with the database driver. We will solve with a workaround, but what about next time?

What would the impact be if the data were compromised in some way?	If they find a hole that requires a major upgrade, it would be a disaster. For those applications we have a relaxed SLA with just 1 nine, but a fix would require days if not weeks.
What is the value to the organization?	Those applications bring around 5% of our revenues, but in case of disruption for days it would cost us a lot of money in damages.
What can be done to minimize the exposure to the threat/vulnerability?	Our findings with dependency scans and pen tests can be anything from trivial details in the HTTP headers to serious SQL injections. We can fix anything that is in our code, but we rely heavily on dependencies from third parties and some of the latest releases cannot be integrated anymore.