BUSINESS RULES:

1. Each Robot must be registered in the system before it can execute commands.
2. Each Mission must be defined before commands can be assigned to it.
3. Each Command must be assigned to exactly one Robot.
4. Each Command must belong to exactly one Mission.
5. A Robot can execute multiple Commands over time.
6. A Mission can contain multiple Commands.
7. Commands have a status that tracks their execution state (PENDING, EXECUTING, COMPLETED, FAILED).
8. Robots have a status indicating their availability (ACTIVE, MAINTENANCE, OFFLINE).
9. Each Command has a priority level to determine execution order.
10. Command execution timestamps are recorded for audit purposes.

REPORT:

1. INTRODUCTION

The system works as follows: robots perform commands which are associated with missions. The purpose of the database is to keep track of how the plan is working and gathering data for future improvements.

2. DESIGN DECISIONS

I chose those 3 tables (Robots, Commands, Missions) in order not to overcomplicate our problem. They work together really well and are completely sufficient for the problem. I also chose 1:many relationship between Robots and Commands because I designed it this way that each robot has its own set of actions (even if it does the same as other robots). Data types were meant to be compact and full of insights

3. BUSINESS LOGIC

The system could gather data in the background if it was implemented in a real company. It would probably be some kind of a factory, allowing robots and machines to improve over time.

4. POTENTIAL EXTENSIONS

Over time Logs table could be added to “outsource” already existing tables if they become too large. Also some backup storing would not hurt.

5. CONCLUSION

To sum up, my design is really practical and attempts not to be an overkill. It has everything it needs for the given problem to be fully solved as well as it has a great potential to be extended and updated in the future.