PROG3070 – ADVANCED SQL

PROJECT – MANUFACTURING SCENARIO

OVERVIEW

This assignment integrates many of the concepts presented in this course into one fairly realistic solution. The goal of the solution is to automate a Kanban-based workflow.

The manufacturer builds a fog lamp assembly for automobiles. The assembly is made up of a harness, a reflector, a housing, a lens, a bulb and a bezel. Each of these parts is in its bin, and an employee takes one of each of these parts and puts it together. Once completed, the fog lamp is placed on a conveyor belt to the testing and packaging department. The finished lamp has a barcode attached identifying the assembly station that produced it.

The workflow is focused on minimizing the time it takes to assemble the foglamps.

OBJECTIVES

- Implement Views in SQL to simplify queries, and to improve data retrieval performance.
- Discuss the appropriate use of indices in SQL, and demonstrate how indices improve data retrieval times
- Describe the purpose of stored procedures, and demonstrate how they are created and modified
- Implement at least one data visualization technology to show how large amounts of data can be represented graphically.
- Discuss various techniques in optimizing database performance.
- Identify security considerations when creating a database solution.

ACADEMIC INTEGRITY AND LATE PENALTIES

- Please refer to the SET Policies document regarding Academic Integrity Information
- Please refer to the SET Policies document regarding Late Policy

EVALUATION

Item	Marks
Milestone 1	15
Milestone 2	15
Scalable database design including all triggers, views, stored procs, etc.	10
All scripts for the database	5
Workstation simulation program	10
Workstation Andon display program	10
Runner workstation display	10
Assembly line display program	5
Configuration tool	5
Working simulation	10
Documented Instructions	5
Total	**100**

REQUIREMENTS

Submission in a group of 2 or individual

PROJECT BACKGROUND AND FUNCTIONALITY DESCRIPTION

Your customer is asking you to create a database-backed simulation to support and test a simple electronic Kanban solution.

The customer builds a fog lamp assembly for automobiles. The assembly is made up of a harness, a reflector, a housing, a lens, a bulb and a bezel. Each of these parts is in its own bin, and an employee takes one of each of these parts and puts it together. Once completed, the fog lamp is placed on a tray with separators. When full, the tray is sent to the testing department via conveyor belt.

Each bin has the following capacities, and the bins are full at the start of the process:

- Harness: 55
- Reflector: 35
- Housing: 24
- Lens: 40
- Bulb: 60
- Bezel: 75

These capacities may vary in the future.

As the assembler takes parts from the bins, a sensor at each bin reduces the count in the appropriate bin by 1. When there are only 5 parts left in a bin, a "runner" is notified via some application. This notification identifies the assembly location and the part(s) that is running low.

Every 5 minutes, the "runner" picks up a set of new bins of the specific parts and locations then goes to the stations and replaces them with the old ones. New bins have the starting quantities outlined above. Any remaining parts are placed on top of the stock in the new bin. The return time to/from the stock room fits into the 5-minute span mentioned above. A button is near each bin for the runner to press once a bin is replaced.

The assembly area dedicated to fog lamps has 3 assembly stations, though the customer projects this might need to be expanded in the near future. An experienced worker can assemble a completed fog lamp in 60 seconds, +/- 10%. New employees in this position can take about 50% longer. Very experienced (super) workers can produce a new lamp in 15% less time. Average defect rates for lamps based on the workers' skill levels are to be modelled as: New/Rookie 0.85%, Experienced/Normal 0.5%, and V. Experienced/Super 0.15%.

The finished lamps are tested once the lamp is finished. The failed lamps are discarded, and the good lamps are packaged.

The stock room has an effectively unlimited supply of new parts bins for the runner to bring to the assembly stations, so we do not need to model it for this project.

Build a database-based solution to support an electronic version of this system. Use triggers, functions, views and stored procedures as is appropriate. In addition, you will need to provide visualizations for the workstations, and the assembly area as a whole.

SPECIFIC REQUIREMENTS:

1) A database schema must be developed and implemented representing the database used to support this solution. This must include a configuration table to allow for full configuration of parameters for the simulation. One important parameter will be to control the time scale of the simulation (ie how you represent an hour or minute of real time).
2) Stored procedures, triggers, views and functions should be designed (as appropriate) and created to support all possible activities from within the database.
3) A C# program must be created to represent a single workstation. The purpose of this program is to provide the simulated timing surrounding the creation of a fog lamp. This program may run multiple times simultaneously. For example, three running at the same time would represent the current manual system.
4) A C# program must be created to display a graphic representation of the part counts and status of a single workstation (Andon display) in real time. This visualization must also show a signal for the "runner" to replenish part counts. The number of instances of this

program must match the number of workstations running. This visualization must be separate from the simulation in part 3 above.

5) A C# program to display the status of the entire assembly line in real time. This would be in the form of a conventional Kanban display with Order amount, in process amount, number produced and yield. You may also show other information that you see fit to display. The order amount is somewhat arbitrary for the simulation and can be part of the configuration. In reality, this order amount would be provided by another system. You do not have to implement the other system.
6) A C# program to display station status for the runner. This should identify the bins that need replacement. User interface is important.
7) A C# program to maintain the configuration table data. The configuration editor should be scalable and can be as simple as an interactive data grid.
8) You must design the system so multiple computers can be used. For example, each workstation, the assembly system, the runner application, and the database may all be on separate computers.
9) The simulations must be separate from the other applications. The intention is to just switch out the simulation and have the system working with the live assembly line with only configuration changes.

(continued)

MILESTONE 1

- Database design complete
- Implementation of Configuration table complete
- Configuration tool complete
- Submit diagrams for database design, script for configuration table and solution for Configuration tool

MILESTONE 2

- Database implemented with any changes recommended from Milestone 1
- Simulation complete
- Does not include visualizations
- Submit scripts for the database, and solutions for Configuration tool, and workstation simulation

DEMONSTRATION

- Full demonstration of system

FINAL SUBMISSION

Please submit all of the following in a single zip folder:

- All scripts for the database (including initial data for configuration table)
- All source code for the five programs
    - Workstation simulation
    - Workstation Andon
    - Runner display
    - Assembly line Kanban
    - Configuration tool

A readme file with instruction on setting up and running the simulation must be included.