Système de monitoring avec génération et gestion d'alertes

Version 1 du 20 septembre 2024

UE Système de gestion de bases de données : module 3
AA Système de gestion de bases de données

3ème Bachelier en Informatique et Systèmes
Finalités ”Industrielle” et ”Réseaux et Télécommunications”

Samuel Hiard samuel.hiard@hepl.be
François Caprasse francois.caprasse@hepl.be

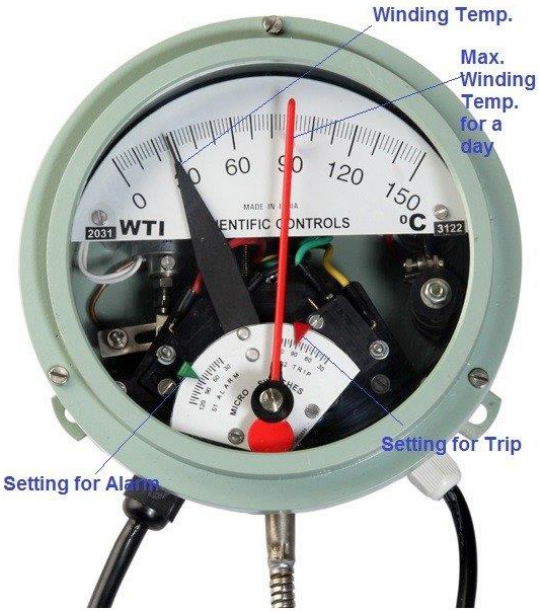

Table des matières

1. Préambule

Vous trouverez dans ce document le contexte général du système informatique qui devra être réalisé. Il s'agit d'une première version temporaire. Ce document détaille les différentes fonctionnalités demandées. Il doit être considéré comme votre "contrat de travail". Il permettra de mieux appréhender l'organisation du travail de l'année et sa planification au travers des différentes semaines.

Les modalités d'évaluation se trouvent à la fin de cet énoncé reprenant les explications quant à la construction de la cote finale. Il convient d'en prendre connaissance. Elles vous engagent pour toute la durée de l'année scolaire.

Les données utilisées proviennent de https://www.kaggle.com/.

2. Contexte général

Une société spécialisée dans la transformation de courant électrique fait appel à vous pour informatiser la gestion de leurs alarmes.

En fonction des données observées, on peut en déterminer si le comportement est normal ou non, et déclencher une alarme ou un arrêt d'urgence le cas échéant.

2.1 Les données

Les données ont été récoltées depuis l'adresse https://www.kaggle.com/datasets/sreshta140/ai-transformer-monitoring?select=Overview.csv que l'on vous recommande de consulter car cette page décrit en détail les données mais, par facilité, nous allons également les présenter ci-après.

Les données qui nous intéressent sont centralisées dans le fichier « Overview.csv » qui contient 8 champs.
La signalétique des champs est décrite à la page suivante.

Nom	Explications
DeviceTimeStamp	Un timestamp avec une date et une heure (attention, les valeurs ne sont pas uniques).
OTI	Oil Temperature Indicator
WTI	Winding Temperature Indicator
ATI	Ambient Temperature Indicator
OLI	Oil Level Indicator
OTI_A	Oil Temperature Indicator Alarm
OTI_T	Oil Temperature Indicator Trip
MOG_A	Magnetic oil gauge indicator

3. Consignes générales de développement

Cette section vous donne des indications générales sur ce que vous pouvez faire, ce que vous ne pouvez pas faire, ce à quoi il faut apporter une attention particulière, etc. lorsque vous réaliserez les différentes fonctionnalités demandées à la section 4, page 5.

3.1 Personnalisation des fonctionnalités

Un certain nombre de points ont été simplifiés par rapport à la réalité. Certains points ont également été volontairement maintenus dans le vague ou l'obscur. L'imagination est la bienvenue pour donner à vos solutions un cachet plus personnel, plus réaliste, plus professionnel. Cependant, pour vous éviter de perdre votre temps à réaliser des options présentant peu d'intérêt, il vous est vivement recommandé de prendre conseil auprès du professeur concerné par le projet.

3.2 Exploitation des atouts de SQL et focus sur SGBD

Lorsque vous écrivez votre code, que ce soit PL/SQL, applicatif ou SQL, gardez bien à l'esprit qu'on est dans le monde des bases de données et donc qu'on essaie de trouver des solutions qui exploitent au maximum le langage SQL et tout ce que la base de données peut nous offrir (optimisation automatique des requêtes, index, etc.). Par conséquent on vous demande d'éviter à tout prix des solutions de type procédural comme si le langage SQL n'existait pas. Ci-dessous se trouve un exemple de pseudo-code pour illustrer ce point où clairement il serait nettement plus profitable de migrer le test présent dans le IF dans la clause WHERE du SELECT et laisser ainsi le système de gestion de base de données faire son travail.

```
SELECT * BULK COLLECT INTO montableau FROM matable;

FOR i IN 1..montableau.COUNT LOOP

IF montableau(i).monchamp = mavaleur THEN

....

END IF;

END LOOP;
```

3.3 Couche ORM ou pas ?

Quel que soit le langage de programmation choisi ou le framework web choisi, nous vous demandons de n'utiliser aucun intermédiaire de type ORM (Object Relational Mapping) entre votre code et les bases de données Oracle comme Java Data Objects (JDO), Java Persistence API (JPA) dont Hibernate, ActiveRecord (Ruby), DataMapper (Ruby), Linq, etc. Par conséquent, il est nécessaire que vous écriviez vous-même les requêtes SQL que vous soumettrez à votre BD.

3.4 Choix du langage de programmation

Le choix du langage de programmation à utiliser se pose lors de la réalisation de certaines fonctionnalités. Ce choix est libre mais nous attirons votre attention sur les points suivants :

- Le choix supporté par le cours est d'utiliser ORDS pour accéder à votre BD Oracle. Le langage supporté par le cours est Java.
- Les autres choix ne sont pas supportés par le cours. Cela signifie donc qu'en cas de problème, nous ne pourrons vous apporter qu'une aide limitée (voire aucune aide, selon les cas). Attention aux problèmes liés au manque de mises à jour et de documentation des implémentations d'accès à Oracle. Bien que ce soit une très bonne idée d'explorer d'autres langages que ceux enseignés au sein de l'école, la perte de temps qui peut résulter de ces choix nous pousse fortement à déconseiller leur utilisation. Néanmoins, nous désirons laisser le choix du langage libre.

Attention, pour certaines fonctionnalités, nous imposons l'appel de procédures stockées. On vous demande de prêter attention à l'existence d'un support pour l'utilisation des BLOBs qui sont obligatoires.

3.5 Choix du SGBD

Pour que ce soit plus facile pour vous, nous vous avons fourni la machine virtuelle Oracle, provenant de https://www.oracle.com/database/technologies/databaseappdev-vm.html et qui vient, pré-installée, avec Oracle database 19, SQL Developer, ORDS, et APEX.

Vous êtes libres d'utiliser votre propre installation d'oracle, mais nous vous rappelons qu'Oracle XE ne contient pas, par défaut, ni ORDS, ni APEX, et que l'installation de ces deux modules (en particulier le dernier) est loin d'être simple.

4. Réalisations attendues

4.1 Phase 1 : Création d'une table externe

Les données, telles que téléchargées, sont au format CSV (comma-separated values), ce qui est très bien pour Excel, mais beaucoup moins pour Oracle.

Il faudra donc intégrer ces fichiers dans une table externe (EXTERNAL TABLE) afin de pouvoir effectuer des requêtes dessus. Bien sûr, pour que ces fichiers soit visible depuis Oracle, il faut qu'ils soient dans un répertoire (DIRECTORY au sens d'Oracle).

Techniques attendues : Répertoires, tables externes.

4.2 Phase 2 : Récupération des données

Vous développerez une application (Java, C++, Python, … au choix) qui récupèrera les données depuis la base de données Oracle. La liaison s'effectuera via ORDS en utilisant le verbe GET, et vous devrez pouvoir, dans un premier temps, récupérer toutes les informations correspondant à 10 lignes sélectionnées au hasard depuis une table interne qui aura été préalablement copiée à partir des données de la table externe.

Techniques attendues : REST, ORDS.

4.3 Phase 3 : Affichage graphique

Les données ainsi collectées seront affichées dans votre application, dans laquelle vous aurez bien sûr développé une GUI.

Nous allons supposer que les 10 lignes prises au hasard représentent les 10 dernières secondes observées.

Techniques attendues : JFreeChart.

4.4 Phase 4 : Sauvegarde d'un snapshot du graphique

La GUI devra permettre la prise d'un instantané du graphique en cours d'affichage. Cet instantané, sous forme d'une image PNG, sera sauvegardée sur la base de données. On utilisera toujours ORDS pour le transfert vers la base de données, avec le verbe POST. Le contenu de l'image sera d'abord encodé en base 64 dans la GUI, puis décodé par Oracle avant d'être sauvegardé dans un BLOB. La date et l'heure de la sauvegarde sont également associées à l'image.

Techniques attendues : CLOBs, BLOBs, Base64, ORDS, APEX

4.5 Phase 5 : Génération automatique

Vous créerez une table maintenant en permanence les 10 derniers relevés. Ces relevés seront générés automatiquement toutes les secondes à l'aide d'un job. Pour être très clair : chaque seconde, le relevé le plus ancien sera supprimé puis un nouveau relevé sera généré. A vous de vous arranger pour conserver l'ordre des relevés. Cette génération se fera à l'aide d'un job qui tirera au hasard des valeurs en fonction d'une moyenne et d'un écart-type (un par attribut) qui seront maintenus dans une table se trouvant dans une base de données distante (accessible par Db Link). Le visualisateur doit également être mis à jour afin de pouvoir afficher un déroulé en temps réel (delta d'affichage = 1 seconde) des 10 derniers relevés.

Note : Pour la generation, vous pouvez utiliser la transformation de Box-Muller. Soit une moyenne *mean* et une distribution standard *stdev* ainsi que 2 valeurs aléatoires uniformes *rand1* et *rand2* générées par `dbms_random.value`, vous pouvez produire une valeur aléatoire respectant la distribution de probabilité souhaitée en utilisant la formule :

next_value := mean + std * SQRT(-2 * LN(rand1)) * COS(2 * 3.14159265359 * rand2);

Techniques attendues : Jobs, DBLinks.

4.6 Phase 6 : Analyse de données

Plutôt que d'afficher simplement les valeurs observées, on vous demandera de réaliser une analyse de données permettant de déterminer, à partir de la table externe, à partir de quel moment l'alarme, le trip ou l'indicator se déclenchent, et de répercuter ces informations dans l'interface graphique lorsque les valeurs générées dépassent le(s) seuil(s) en question.

Techniques attendues : Exploration/Analyse des données.

5. Organisation pratique et évaluations

5.1 Première session (Janvier)

- L'évaluation du cours est une évaluation orale à 100%.
- Lors de cette évaluation, vous présenterez votre travail, en faisant une petite démonstration. Nous vous interrogerons à la fois sur le code source, mais aussi sur les concepts théoriques. Il n'est pas impossible que l'on vous demande une (petite) modification de code lors de la présentation.
- Le travail est prévu pour deux, mais chaque étudiant doit être capable de répondre à toute question du titulaire portant sur l'application présentée ou la matière théorique et pratique liée à l'application
- Les étudiants sont libres d'organiser et de répartir leur travail comme ils le souhaitent en tenant compte des règles indiquées ici.

5.2 Deuxième session

L'examen de seconde session se fera dans les mêmes conditions que celui en première session (même énoncé, mêmes fonctionnalités).
Les groupes peuvent être changés si vous le désirez.
L'évaluation sera reprise de zéro et compte tenu du délai supplémentaire, des commentaires et corrections donnés lors de l'évaluation de première session, il est possible de ne pas obtenir autant de points pour certaines réalisations qu'en janvier. La cote de première session n'est pas conservée si celle de seconde session est moins bonne.

Et surtout, amusez-vous bien !