

Sistema de Ensino 100% online

Superior em Tecnologia em Análise e desenvolvimento de sistemas

FABIO ARAUJO SILVA

**Sistema FAZENTECH**

SANTO ANDRE / SP

2020

fabio araujo silva

**Sistema FAZENTECH**



SANTO ANDRE / SP

2020

SUMÁRIO

[1 INTRODUÇÃO](#_Toc48729576) 4

[2 DESENVOLVIMENTO.](#_Toc48729577) 5

[2.1 ANÁLISE ORIENTADA A OBJETOS](#_Toc48729578) ..5

[2.2 DESENVOLVIMENTO DE BANCO DE DADOS](#_Toc48729579) 13

[2.3 LINGUAGENS DE PROGRAMAÇÃO](#_Toc48729580) 18

[3 CONCLUSÃO 21](#_Toc48729581)

# INTRODUÇÃO

Esse tema proposto, a produção textual a seguir, tem como objetivo desenvolver um sistema para melhor gerenciar uma Fazenda. Onde o objetivo principal é focado em: preservação e otimização dos recursos, geração de produtos e serviços inovadores e proporcionar satisfação e equilíbrio aos envolvidos. Onde deve ser implantado sistemas informatizados e tecnologias de ponta.

O estudo de caso em questão, coloca o acadêmico para desenvolver um sistema de informação de gerenciamento para uma fazenda. A intenção é fazer com que esse sistema possa oferecer dados e informações precisas e detalhadas para que o gestor possa tomar suas decisões de forma assertiva e com mais eficácia. Nesse sentido, iremos apresentar todo o banco de dados juntamente com seu script e a as possíveis soluções para a situação problemática da fazenda, que gira em torna da perda de leite.

# DESENVOLVIMENTO.

## ANÁLISE ORIENTADA A OBJETOS I

Todo e qualquer desenvolvimento de software possui um ciclo de vida que é baseado em etapas e estágios diferenciados, assim são utilizados também conceitos soltos e abstratos com finalidade de divisionais atividades que ocorrem em cada etapa de desenvolvimento. Dentre as etapas, podemos citar como participantes, processos como: planejamento, requerimentos, design, formato, dentre outros.

O desenvolvimento de um software é algo bem demorado e que exige bastante do desenvolvedor e programador, principalmente em termos de conhecimento e desenvolvimento do que é solicitado para tal necessidade do cliente. Assim, o sistema deve estar de acordo com a finalidade pretendida.

Ao logo da etapa de análise do sistema ou também conhecido como análise orientada a tais objetos para desenvolvimento de software, os requisitos do sistema devem ser pautados e determinados, as classes utilizadas devem ser identificadas e assim como também os relacionamentos e relações entre tais classes que foram implementadas.

Existem três técnicas de análise de sistemas que são utilizadas comumente em conjunto para com a análise orientada para com objetos, são assim: modelagem de objetos, modelagem dinâmica, e a modelagem funcional. Essas técnicas devem funcionar em auxilio uma da outra, contribuindo para uma análise eficiente e de qualidade.

A primeira técnica que é fundamentada na modelagem de objetos procura desenvolver uma estrutura estática para o sistema de software em termos de objetos. Esta identifica de forma precisa os objetos, as classes que os objetos podem ser alocados e agrupados e assim também os relacionamentos entre esses objetos. Nessa técnica ainda é capaz também de identificar todos os principais atributos e operações para com cada classe.

Após a conclusão da etapa voltado para o desenvolvimento de um comportamento estático para o sistema, assim é analisado o comportamento deste em relação ao tempo e também as mudanças externas que ocorrem precisam de uma examinação sucinta, assim temos o objetivo da modelagem dinâmica.

A modelagem dinâmica é um conceito bem difundido na área de sistema, que tem por finalidade descrever como um objeto individual responde a determinados eventos, sejam esses internos ou externos. Onde, os eventos internos são acionados por um outro objeto qualquer e os eventos externos que são acionados através de situações do mundo exterior.

Nesse sentido, observado o caso da situação geradora do sistema FazenTECH foi realizada a etapa voltada para modelagem de atividades de análise de sistemas foi através de uma ferramenta chamada de CASE para modelagens, onde contempla também a UML. Então, deve-se levar em consideração todas as funcionalidades para realizar o processo de criação animal e também voltado para o planejamento do plantio nas mais diferentes culturas da fazenda em estudo.

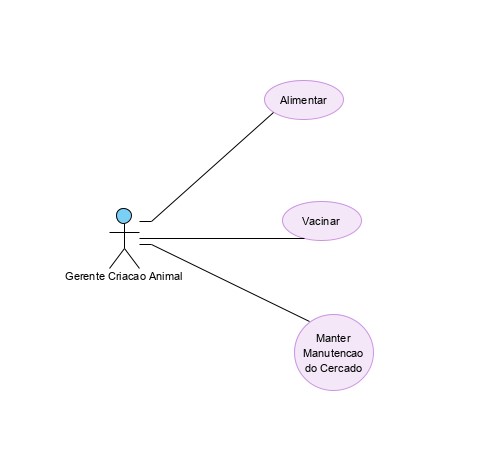
Portanto, para atender de forma concisa e continua o domínio de criação animal e também o planejamento elaborado para o plantio do sistema FazenTECH, é evidenciado a seguir o modelo de Casos de Uso, assim como também o modelo de Classes e por último e não menos importante, o Diagrama de Máquina de Estados para a classe determinada “Plantio”, relatando sobre estes em cada uma das fases desse artigo.

1. **O Diagrama de Use Cases:**

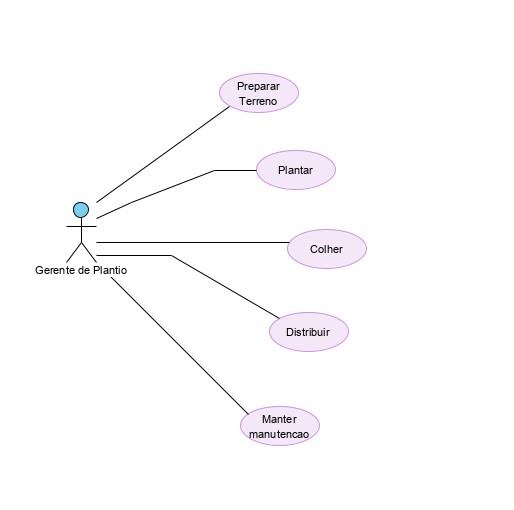
O objetivo e finalidade principal desse diagrama de caso de uso é registrar o aspecto dinâmico de um determinado sistema. Então, assim essa definição se torna muito ampla e genérica, não deixando claro o propósito destes, visto que ainda existe quatro diagramas distintos, tais como: o de atividade, o de sequência, o de colaboração e o gráfico de estado, que possuem propósito semelhantes e até mesmo iguais. Assim, existe alguns fatores e aspectos específicos que trazer a distinção para com os demais tipos de diagramas.

Os diagramas de caso de uso são aqueles utilizados para fazer uma reunião dos requisitos fundamentais e essenciais para um sistema, isso inclui influências tanto internas como externas. Dentre esses requisitos, podemos até mesmo encontrar o de design que é bem importante para tal processo. Assim, quando um sistema é analisado, este busca reunião em sua amplitude suas funcionalidades, com isso os casos de uso são preparados e todos os atores são identificados.

Assim, quando a tarefa inicial é concluída, os diagramas de caso de uso são modelados para se apresentar a visão externa deste:



**Figura 1: Diagrama de Caso de Uso “Criação Animal”**



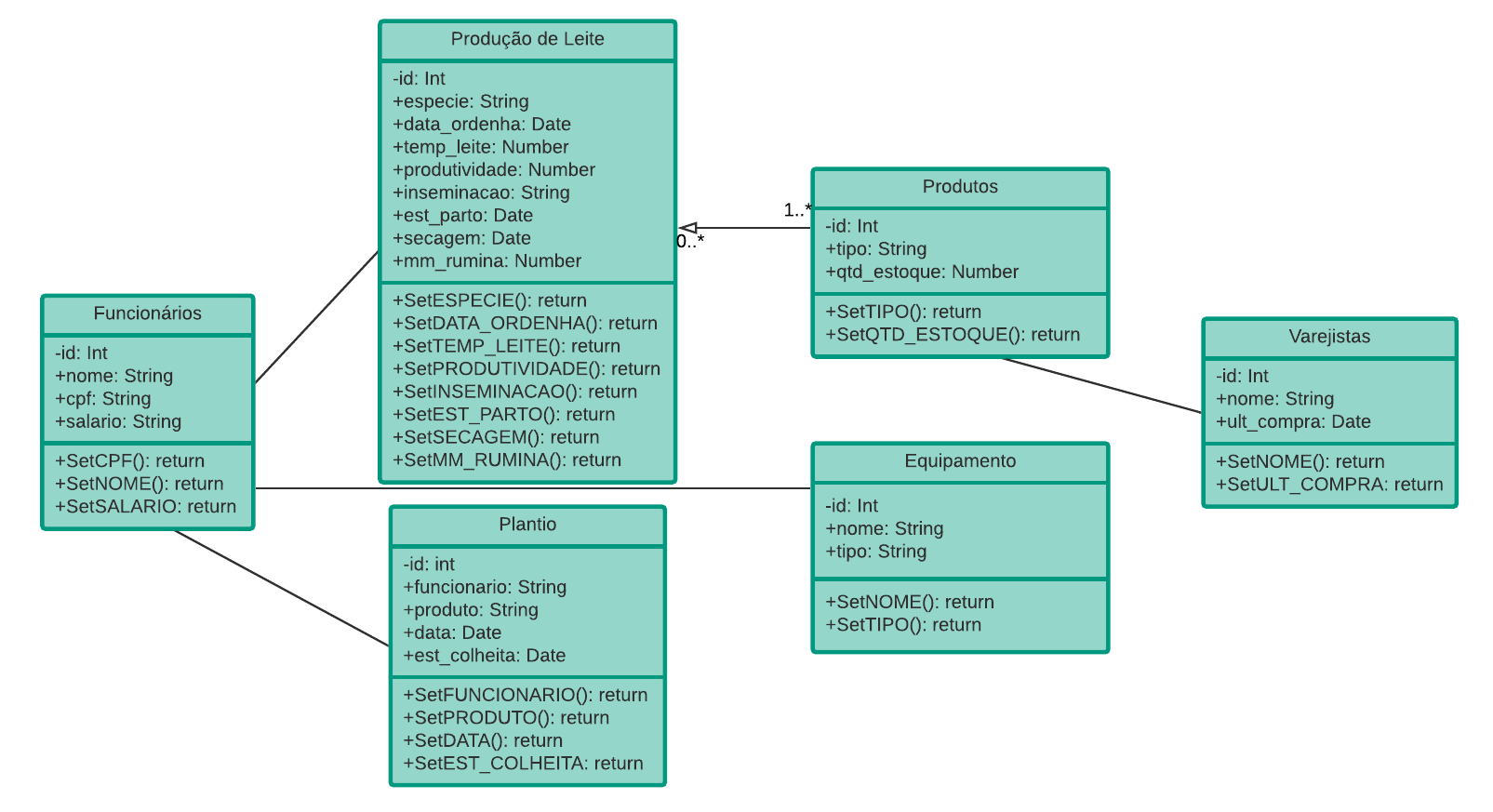
**Figura 2: Diagrama de Caso de Uso “Plantio”**

1. **O Modelo de Classe:**

O modelo de classes tem objetivo de descrever e estruturar os objetos de um sistema, assim os diagramas de classes dizem respeito, evidenciando o modelo de classe. Nesse sentido, o diagrama de classes se trata de um diagrama estático, ou seja, parado, onde este apresenta uma visão estática de um aplicativo. O diagrama de classes busca visualizar, documentar, descrever e entender os mais diferentes aspectos de um sistema especifico, assim como também a construção de um código que será executável através de um aplicativo desenvolvido em software.

O diagrama de classes detalha todas as características e atributos, assim como também operações de uma determinada classe e assim também restrições que são impostas ao sistema. Os diagramas de classe são amplamente utilizados para modelagem de sistemas orientados para objetos, pois são os únicos diagramas UML, onde podem ser mapeados com uma linguagem orientada e focada para objetos.

O diagrama de classes evidencia um complexo ou conjunto de coleção de classes, interfaces, colaborações, associações e também restrições, sendo assim também conhecido como diagrama estrutural. Vejamos:

**Figura 3: Diagrame de Classe**

1. **O Diagrama de Máquina de Estados:**

O diagrama de máquina de estado tem por objetivo e finalidade estruturar e modelar como se dá o comportamento de apenas um objeto. Assim, dentro deste se especifica a sequencia de eventos que o objeto passa ao longo de toda sua vida útil para com resposta a esse evento.

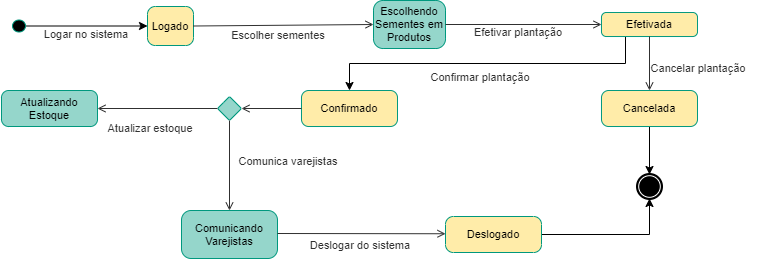
Esse diagrama se trata de um dos cinco diagramas UML utilizados para estruturar e modelar a natureza dinâmica de um sistema, assim consegue se definir todos os possiveis estados de um objeto ao longo de todo o seu período de vida, onde esses estados são modificados e alterados por eventos. A modelação principal é voltado para sistemas reativos, onde estes podem ser definidos como sistemas que responde a disversos eventos, tanto internos como externos.

Esse diagrama tem como uma de suas atividades também descrever e desenhar o fluxo de controle que será utilizado de um estado para outro. Os estados são compreendidos e entendidos como uma condição determinada que um objeto se encontra, existe e muda assim que um evento é acionado. Então, é entendido como foco principal desse diagrama de máquina de estado a modelação do tempo de vida de um objeto desde a origem dele até a sua finalização.

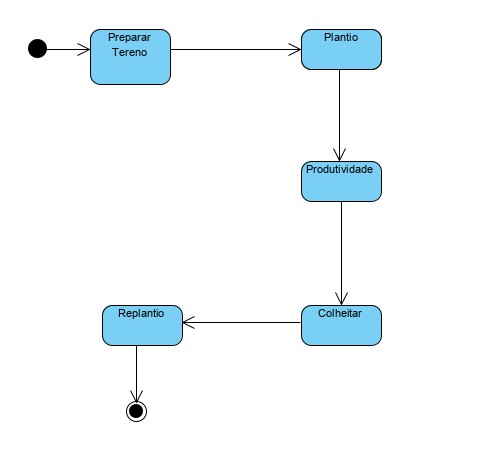
Dentro do diagrama, um estado é configurado e identificado por um retângulo arredondado, onde vem seu nome descrito dentro deste. Assim, o estado inicial é denotado e descrito por um circulo preto que é preenchido e que possui como rôtulo seu nome. Já o estado final, é identificado por um circulo com um ponto dentro que também é rotulado pelo um nome especifico.

Podemos assim, observar o diagrama de máquina de estados para o sistema desenvolvido para a FazenTECH:

**Figura 4: Diagrama de Máquina de Estados**

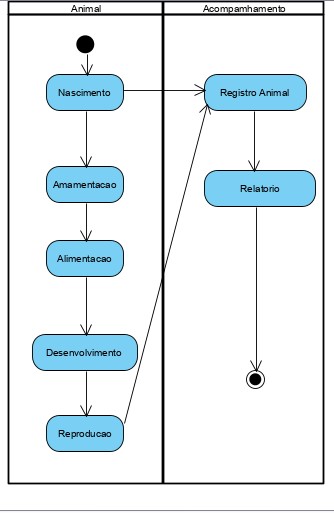


**Figura 5: Diagrama de Máquina de Estados Plantio**



### Diagrama de atividades

O diagrama de atividade ajuda a unir as pessoas das áreas de negócios e de desenvolvimento de uma organização para entender o mesmo processo e comportamento. Para criar um diagrama de atividade, é necessário um conjunto de símbolos especiais, incluindo aqueles para dar partida, encerrar, fundir ou receber etapas no fluxo. Diagrama de atividade oferece uma série de benefícios, tais como: Demonstrar a lógica de um algoritmo, descrever as etapas realizadas em um caso de uso UML, ilustrar um processo de negócio ou fluxo de trabalho entre usuários e o sistema, simplificar e melhorar qualquer processo ao esclarecer casos de uso complicados, modelar elementos de arquitetura de *software*, como método, função e operação.

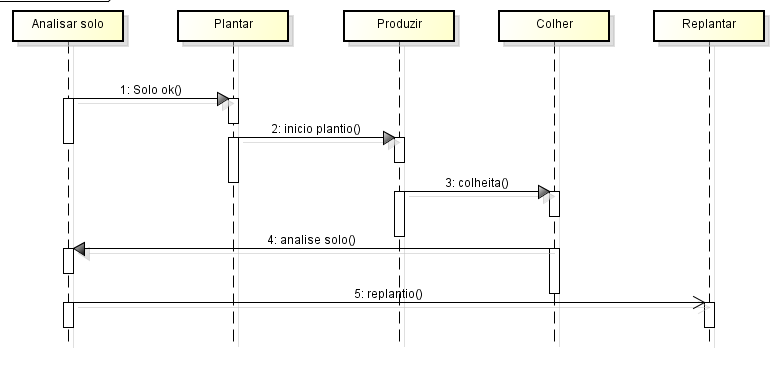


**Figura 5: Diagrama de Atividades**

1. **Diagrama de sequencia**

O diagrama de sequência é uma espécie de diagrama de interação, pois descreve como, e em qual ordem, um grupo de objetos trabalha em conjunto. Estes diagramas são usados para entender as necessidades de um novo sistema ou para documentar um processo existente. Diagramas de sequência são conhecidos como diagramas de eventos ou cenários de eventos. Existem dois tipos de diagramas de sequência: Diagramas UML e diagramas baseados em código. O baseado em código é proveniente de código de programação, o *software* de diagramação UML do *Lucidchart* contém todas as formas e recursos necessários

**Figura 6: Diagrama de sequência**



## DESENVOLVIMENTO DE BANCO DE DADOS

O Banco de Dados se trata de um aplicativo especifico e separado que armazena um conjunto de dados, esses dados serão utilizados pelo gestor quando necessário. Assim, cada banco de dados possui uma ou mais APIs diferentes que servem para criar, gerenciar, acessar, pesquisar e até mesmo replicar dados que estão ali embutidos.

Pode-se fazer outros tipos de armazenamento de dados, tais como arquivos no sistema ou grandes tabelas hash na memória, mas assim a busca e a gravação que é um dos fatores primordiais não seriam tão rápidos e fáceis como tais esses tipos de sistemas. Em nossa atualidade, são utilizados sistemas de gerenciamento de banco de dados relacional com objetivo de armazenar e gerir um volume muito grande de dados. Um banco de dados relacional se trata de um sistema de armazenamento de dados em tabelas diferentes e as relações são sempre estabelecidas utilizando chaves conhecidas, tais como chaves estrangeiras.

O sistema criado para FazenTECH necessita de armazenar uma série de dados e informações que são importantes e relevantes para o negócio, tais como informações pessoais dos usuários, fornecedores, materiais, clientes, produtos, dentre outros. Esse sistema precisa armazenar a quantidade máxima de dados, objetivando assim, fornece o suporte fundamental para tal negócio.

Na análise e desenvolvimento de sistemas existem diversos banco de dados a conhecer: MySQL que é um sistema de gerenciamento de banco de dados do tipo SQL relacional.

Os comandos de Data Definition Language (DDL) são também extremamente importantes, pois criam, manipulam e monificam objetos, assim como usuários, banco de dados, tabelas, esquemas, colunas, visualizações, procedimentos, funções armazenadas. Já os comandos do DML possuem como finalidade de inserir, excluir, atualizar, e até mesmo mesclar dados para com as tabelas, assim os comandos DQL possuem como função selecionar os registros de uma, duas ou mais tabelas.

Esses comandos citados, são utilizados para realizar uma série de operações, podendo ser em nível de sessão e conta, tais como: definição de parâmetros, iniciar transações e até mesmo inicializar variáveis. A seguir será expresso um simples script SQL para auxílio na criação de tabelas que são necessárias para o banco de dados da “fazenda-bd”.

|  |
| --- |
| CREATE DATABASE `fazenda-bd`;  CREATE TABLE `Compras` (  `id` int(11) NOT NULL,  `nome` varchar(250) COLLATE utf8\_unicode\_ci DEFAULT NULL,  `produto` varchar(250) COLLATE utf8\_unicode\_ci DEFAULT NULL,  `qtd` int(13) DEFAULT NULL,  `dt\_compra` date DEFAULT NULL  ) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8\_unicode\_ci;  INSERT INTO `Compras` (`id`, `nome`, `produto`, `qtd`, `dt\_compra`) VALUES  (1, 'Fazenda Santo Andre', 'sementes de girassol', 900, '2020-10-09'),  (2, 'Comunidade Rural Fasendinha', 'sementes de girassol', 1200, '2020-09-16');  CREATE TABLE `Equipamentos` (  `id` int(11) NOT NULL,  `nome` varchar(250) COLLATE utf8\_unicode\_ci DEFAULT NULL,  `tipo` varchar(250) COLLATE utf8\_unicode\_ci DEFAULT NULL  ) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8\_unicode\_ci;  INSERT INTO `Equipamentos` (`id`, `nome`, `tipo`) VALUES  (1, 'colheitadeira', 'motorizado'),  (2, 'ceifadora', 'motorizado');  CREATE TABLE `Funcionarios` (  `id` int(11) NOT NULL,  `nome` varchar(250) COLLATE utf8\_unicode\_ci DEFAULT NULL,  `cpf` varchar(11) COLLATE utf8\_unicode\_ci DEFAULT NULL,  `salario` varchar(15) COLLATE utf8\_unicode\_ci DEFAULT NULL  ) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8\_unicode\_ci;  INSERT INTO `Funcionarios` (`id`, `nome`, `cpf`, `salario`) VALUES  (1, 'Raimundo Silva', '01759702134', '1250'),  (2, 'Antonio Fracisco', '04386902349', '1250'),  (3, 'Claudio Sousa', '79449314539', '3650'),  (4, 'Gildivan Gomes', '73283454294', '3650');  CREATE TABLE `Produc\_Leite` (  `id` int(11) NOT NULL,  `especie` varchar(250) COLLATE utf8\_unicode\_ci DEFAULT NULL,  `data\_ordenha` date DEFAULT NULL,  `temp\_leite` int(11) DEFAULT NULL,  `produtividade` int(11) DEFAULT NULL,  `inseminacao` varchar(3) COLLATE utf8\_unicode\_ci DEFAULT NULL,  `est\_parto` date DEFAULT NULL,  `secagem` date DEFAULT NULL,  `mm\_rumina` int(11) DEFAULT NULL  ) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8\_unicode\_ci;  INSERT INTO `Produc\_Leite` (`id`, `especie`, `data\_ordenha`, `temp\_leite`, `produtividade`, `inseminacao`, `est\_parto`, `secagem`, `mm\_rumina`) VALUES  (1, 'marina', '2020-08-13', 33, 1500, 'nao', '2020-10-25', '2021-05-29', 3500),  (2, 'leiteira', '2020-04-22', 39, 2600, 'nao', '2021-05-13', '2021-11-15', 3500);  CREATE TABLE `Produtos` (  `id` int(11) NOT NULL,  `nome` varchar(250) COLLATE utf8\_unicode\_ci NOT NULL,  `tipo` varchar(250) COLLATE utf8\_unicode\_ci DEFAULT NULL,  `qtd\_estoque` int(11) DEFAULT NULL,  `preco` float DEFAULT NULL  ) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8\_unicode\_ci;  INSERT INTO `Produtos` (`id`, `nome`, `tipo`, `qtd\_estoque`, `preco`) VALUES  (1, 'sementes de girassol', 'sementes', 500, 20),  (2, 'enxada', 'material', 300, 70);  CREATE TABLE `Varejistas` (  `id` int(11) NOT NULL,  `nome` varchar(250) COLLATE utf8\_unicode\_ci DEFAULT NULL,  `ult\_compra` date DEFAULT NULL  ) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8\_unicode\_ci;  INSERT INTO `Varejistas` (`id`, `nome`, `ult\_compra`) VALUES  (1, 'Fazenda Santo Andre', '2020-01-23'),  (2, 'Comunidade Fasendinha', '2020-04-15');  ALTER TABLE `Compras`  ADD PRIMARY KEY (`id`);  ALTER TABLE `Equipamentos`  ADD PRIMARY KEY (`id`);  ALTER TABLE `Funcionarios`  ADD PRIMARY KEY (`id`);  ALTER TABLE `Produc\_Leite`  ADD PRIMARY KEY (`id`);  ALTER TABLE `Produtos`  ADD PRIMARY KEY (`id`);  ALTER TABLE `Varejistas`  ADD PRIMARY KEY (`id`);  ALTER TABLE `Compras`  MODIFY `id` int(11) NOT NULL AUTO\_INCREMENT, AUTO\_INCREMENT=3;  ALTER TABLE `Equipamentos`  MODIFY `id` int(11) NOT NULL AUTO\_INCREMENT, AUTO\_INCREMENT=3;  ALTER TABLE `Funcionarios`  MODIFY `id` int(11) NOT NULL AUTO\_INCREMENT, AUTO\_INCREMENT=5;  ALTER TABLE `Produc\_Leite`  MODIFY `id` int(11) NOT NULL AUTO\_INCREMENT, AUTO\_INCREMENT=3;  ALTER TABLE `Produtos`  MODIFY `id` int(11) NOT NULL AUTO\_INCREMENT, AUTO\_INCREMENT=3;  ALTER TABLE `Varejistas`  MODIFY `id` int(11) NOT NULL AUTO\_INCREMENT, AUTO\_INCREMENT=3;  COMMIT;  [SELECT](https://br140.hostgator.com.br:2083/cpsess6136946034/3rdparty/phpMyAdmin/url.php?url=https://dev.mysql.com/doc/refman/5.5/en/select.html) A.nome, B.nome FROM Compras as A INNER JOIN Varejistas as B on A.nome = B.nome;  [SELECT](https://br140.hostgator.com.br:2083/cpsess6136946034/3rdparty/phpMyAdmin/url.php?url=https://dev.mysql.com/doc/refman/5.5/en/select.html) [COUNT](https://br140.hostgator.com.br:2083/cpsess6136946034/3rdparty/phpMyAdmin/url.php?url=https://dev.mysql.com/doc/refman/5.5/en/group-by-functions.html#function_count)(\*) FROM Funcionarios;  [SELECT](https://br140.hostgator.com.br:2083/cpsess6136946034/3rdparty/phpMyAdmin/url.php?url=https://dev.mysql.com/doc/refman/5.5/en/select.html) [COUNT](https://br140.hostgator.com.br:2083/cpsess6136946034/3rdparty/phpMyAdmin/url.php?url=https://dev.mysql.com/doc/refman/5.5/en/group-by-functions.html#function_count)(DISTINCT nome) FROM Compras;  [SELECT](https://br140.hostgator.com.br:2083/cpsess6136946034/3rdparty/phpMyAdmin/url.php?url=https://dev.mysql.com/doc/refman/5.5/en/select.html) [MAX](https://br140.hostgator.com.br:2083/cpsess6136946034/3rdparty/phpMyAdmin/url.php?url=https://dev.mysql.com/doc/refman/5.5/en/group-by-functions.html#function_max)(salario) FROM Funcionarios; |

**Quadro 1: Script SQL**

## LINGUAGENS DE PROGRAMAÇÃO

Com o objetivo e finalidade de reduzir de forma considerável a quantidade de perdas na produção de leite na fazenda de Lúcia foi implementado um sistema de controle de produção para o negócio de Lúcia como a nomenclatura de FazenTECH a qual utilizou-se uma linguagem chamada de Python. Então, assim a finalidade é implantar uma busca binária dentro desse sistema.

Python é o nome dado a uma grande e poderosa linguagem de programação que possui propósito geral, ou seja, possui diversas finalidades, assim pode-se evidenciar que este pode ser utilizado para desenvolver web, criação de protótipos de software, ciência de dados, dentre várias outras atividades. Assim, para iniciantes, essa linguagem possui uma sintaxe simples e bem fácil de se utilizar, então essa linguagem é a mais apropriada para iniciantes que estão engrenando na programação.

O Python se trata de uma linguagem de programação em plataforma cruzada, o que se entende que este pode ser executado ou rodado em diversas plataformas, tais como: Windows, MacOS, Linux e até mesmo em máquinas virtuais para Java e Net, sendo gratuito e de código aberto. Mesmo que a maioria dos Linux e Mac atuais tenham Python pré-instalado, a versão pode estar desatualizada. Portanto, é sempre uma boa ideia instalar a versão mais atual.

A pesquisa binária se trata de um algoritmo clássico para a ciência da computação, este é muito pautado e evidenciado nessa área. Então, sempre surge em concursos de programação e diversas entrevistas técnicas. Implementar e estruturas a pesquisa binária é algo desafiante e um tanto complexo, mesmo quando se entende bem a sua conceituação. Porém, quando se trata de um curioso ou alguém que tenha habilidades bem especificas, onde sempre deve-se aproveitar os materiais e conteúdo que estão na integra, assim como em bibliotecas para se fazer pesquisas binárias em Python ou até mesmo em qualquer outra linguagem.

A busca binária se trata de um algoritmo altamente eficiente para encontra um item especifico em uma lista toda ordenada, este funciona dividindo rapidamente e de forma eficaz pela metade a porção de toda a lista que deve conter tal item, até assim reduzir as localizações possíveis de apenas uma. A ideia consiste em basicamente dividir repetidamente uma lista previamente ordenada de vacas leiteiras e aplicarmos a busca binária até encontrarmos as vacas que já foram ordenhadas.

Podemos assim, observar a seguir uma consulta binária se utilizando da linguagem Phyton:

|  |
| --- |
| def busca\_binaria(v, p, r, x):    #condição de parada  if p <= r:  q = (p+r) // 2 #buscando o índice do meio do vetor  if x > v[q]:  return busca\_binaria (v, q+1, r, x)  elif x < v[q]:  return busca\_binaria (v, p, q-1, x)  else:  return q #encontrado    return -1 #não encontrado  vacas\_ordenhadas = list (range(1,5000))  vaca = 5001  posicao = busca\_binaria(vacas\_ordenhadas, 0, len(vacas\_ordenhadas)-1, vaca)  if posicao >=0:  print("A vaca %d foi ordenhada e se encontra na posição: %d." % (vaca,posicao))  else:  print("A vaca NÃO foi ordenhada.")  print (vacas\_ordenhadas) |

**Quadro 2: Consulta binária em Phyton**

# CONCLUSÃO

* Após a análise e levantamentos dos dados, o projeto foi iniciado falando um pouco sobre análise orientada a objetos e sua importância. Foi preciso utilizar banco de dados para armazenar os dados sensíveis do sistema FazenTECH e produzir um script para facilitar sua criação.
* Por fim, foi apresentada uma solução para minimizar as perdas da Fazenda no que diz respeito à ordenha de vacas leiteiras. Portanto, esse artigo foi de suma importância e relevância para o desenvolvimento do acadêmico como profissional da área de análise e desenvolvimento de sistemas, pois possibilitou que o mesmo se coloca em prática suas competências para o desenvolvimento e criação de sistemas e tomada de decisão a partir de soluções ideais para determinadas situações.

###### REFERÊNCIAS

FABRIS, P. P. G.; MORAES, E. M.; HISATOMI, M.. **Análise orientada a objetos I**. Londrina: Editora e Distribuidora Educacional S.A., 2018. Disponível em < https://biblioteca-virtual.com/detalhes/livro/1076> Acesso em: 10 out. 2020.

PRESSMAN, R.; MAXIM, B. **Engenharia de software: uma abordagem profissional**. 8 ed. Porto Alegre: AMGH, 2016. Disponível em < https://integrada.minhabiblioteca.com.br/books/9788580555349> Acesso: em 15 out. 2020.

SANTANA, Gisele Alves. **Linguagens de programação e estruturas de dados**. Londrina: Editora e Distribuidora Educacional S.A. 2018.

Mysql .Disponível em: <https://www.techtudo.com.br/artigos/noticia/2012/04/o-que-e-e-como-usar-o-mysql.html> Acesso em: 05 Set. 2020

Programação orientada à objetos. Disponível em: <https://www.devmedia.com.br/principais-conceitos-da-programacao-orientada-a-objetos/32285> Acesso em: 08 Set. 2020

Diagrama de Caso de uso. Disponível em: <https://homepages.dcc.ufmg.br/~amendes/GlossarioUML/glossario/conteudo/caso_de_uso/diagrama_de_caso_de_uso.htm> Acesso em: 05 Set 2020

Diagrama de Classes. Disponível em: <http://www.macoratti.net/net_uml1.htm> Acesso em: 10 Set 2020

O que é UML. Disponível em: <https://www.ateomomento.com.br/diagramas-uml/> Aceso em 12 Set 2020