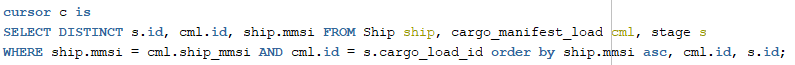
# US 404

A ideia geral desta US é percorrer todas as *stages* de todos os cargos *manifest* *load* de todos os barcos de modo a obter os dias do ano em que o barco esteve parado. Para isso foi criado o cursor demonstrado na figura 1.

Figura 1 - Cursor



A data atual foi colocada numa variável de tipo *timestamp* “dataAtual”, e depois foi retirado o ano a essa data e foi inserida numa variável *integer* “inyear”.

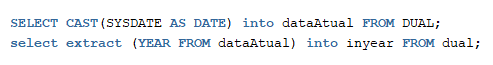


Figura 2 - Data atual e ano

Ao percorrer cada *stage*, podiam existir 3 casos:

1. A *departure date* e a *arrival date* ambas coincidem com o ano atual;
2. A *departure date* seria no ano anterior ao atual, mas a *arrival date* poderia coincidir com o ano corrente;
3. A *departure date* seria no ano atual, mas a *arrival date* poderia ser no ano seguinte ao corrente;

O ano da *departure date* da *stage* a ser percorrida é inserida na variável “departure\_year” e o ano da *arrival date* na “arrival\_year”. No caso 1, apenas era necessário subtrair a *arrival date* pela *departure date*. No caso 2, era necessário subtrair a *arrival date* pelo primeiro dia do ano atual (xxxx/01/01). Concluindo, no caso 3 seria subtrair o último dia do ano atual (xxxx/12/31) pela *departure date*. Cada um destes casos irá nos retornar os dias ativos dessa *stage* do navio.



Figura 3 - Soma de dias ativos do navio em cada stage

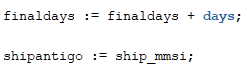
No final de cada *stage*, os dias eram somados na variável *integer* “finaldays” e o *ship* da *stage* era sempre adicionado à variável “shipantigo”.

Figura 4 - Soma de dias úteis do navio

Concluindo, no início de cada ciclo é analisada a variável “shipantigo” e é comparada ao *ship* da *stage* atual. Se ambas forem diferentes, significa que agora estamos a analisar outro navio, ou seja quer dizer que as *stages* daquele navio já terminaram. Logo, subtraímos os dias de um ano por os dias úteis do navio para obter os dias em que ele esteve parado. Depois as outras variáveis são definidas de modo a receber um novo navio para ser analisado.

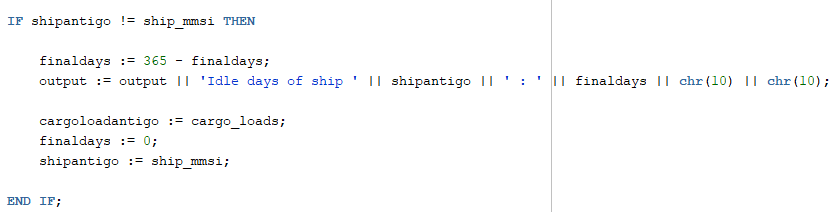


Figura 5 - Output

Na figura 6 está demonstrado o *output* final, que são todos os dias inativos de cada navio no ano atual (2022) e na figura 7 estão demonstradas todas as *stages* do navio “229767000” como prova que os dados estão corretos.

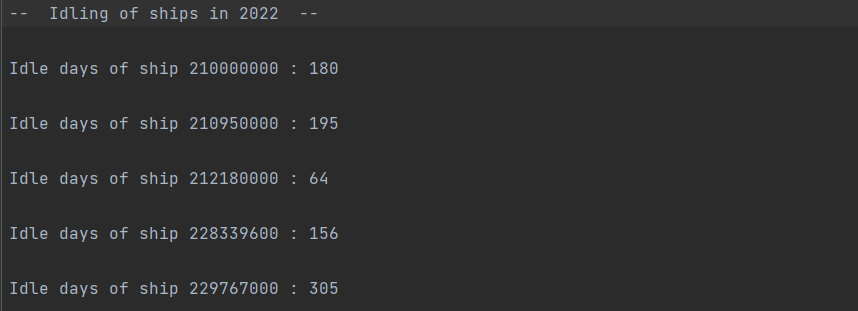


Figura 6 - Output final

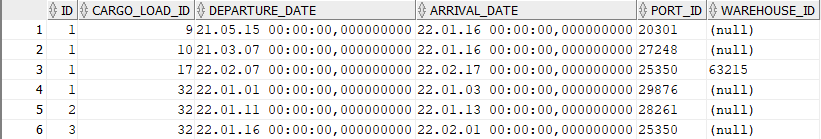


Figura 7 - Stages do navio "229767000"

# US405

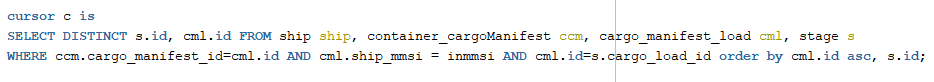
O objetivo desta US é descobrir a taxa de ocupação média de uma *manifest* de um navio num determinado período de tempo. Para isso, foi utilizado o cursor da US404 para calcular a taxa de ocupação média de todas as *stages* porque só depois é possível calcular taxa de toda a *manifest*.

Figura 8 - Cursor

Para a *stage* ter a sua taxa de ocupação média analisada, a *departure date* e a *arrival date* da *stage* têm que estar dentro do período de tempo fornecido. Depois disso, a contagem de containers nessa *stage* fica armazenada na variável *integer* “contcontainer” e esta é usada para calcular a ocupação média dessa *stage*, dividindo pela capacidade do navio. A variável *float* “sumcontcontainer” é a soma das ocupações médias de todas as *stages*, que no fim vão ser divididas pelo número total de *stages* nessa *manifest*, que vai dar a taxa de ocupação média final da *manifest*.

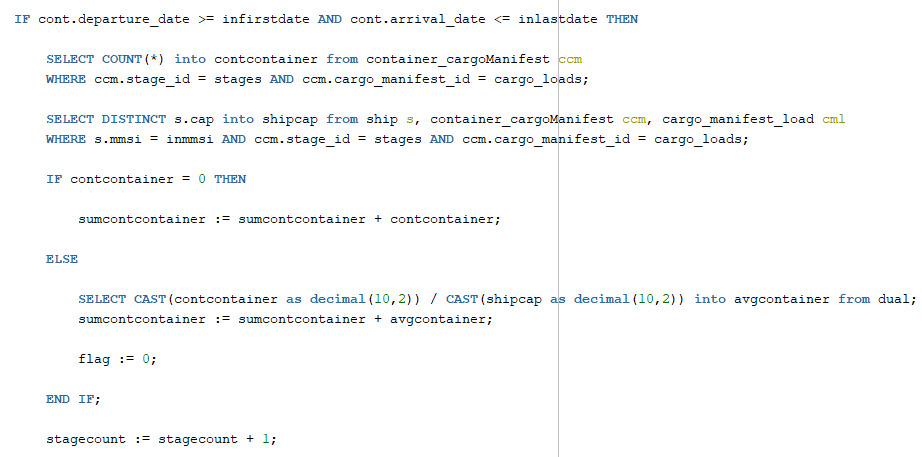


Figura 9 - Calculo da taxa de ocupação média de cada Stage

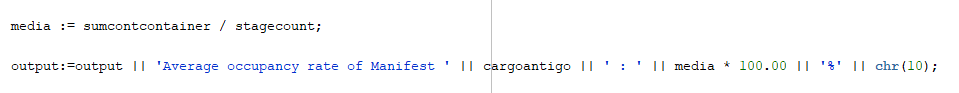
 Tal como dito anteriormente, a variável *float* “media” contém o valor final da taxa de ocupação média da *manifest*, e esta é mostrada após ser detetado a data ter saído dos limites do tempo definido ou por ter sido detetado a última *stage* de uma *manifest*.

Figura 10 - Output da media da manifest

Na figura 11 está demonstrado o *output* final, que são as taxas de ocupação média de todas as *manifests* do navio num determinado tempo e na figura 12 estão demonstrados todos os containers e todas as *stages* do navio “636092932” como prova que os dados estão corretos (capacidade do navio 16 contentores).

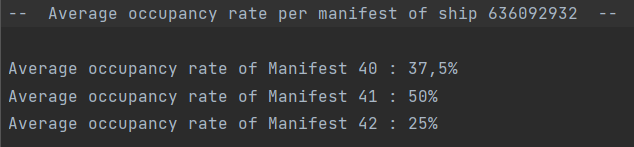
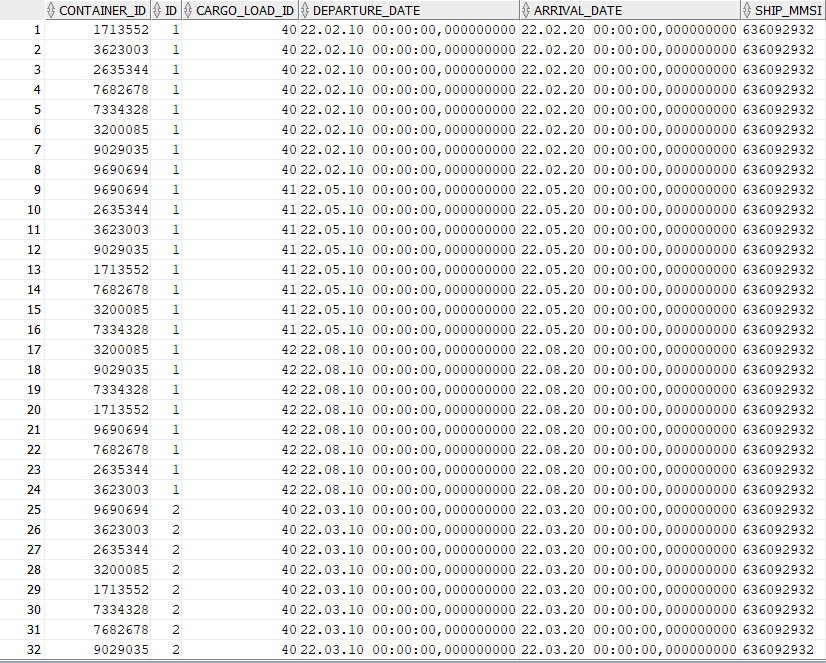


Figura 11 - Output final



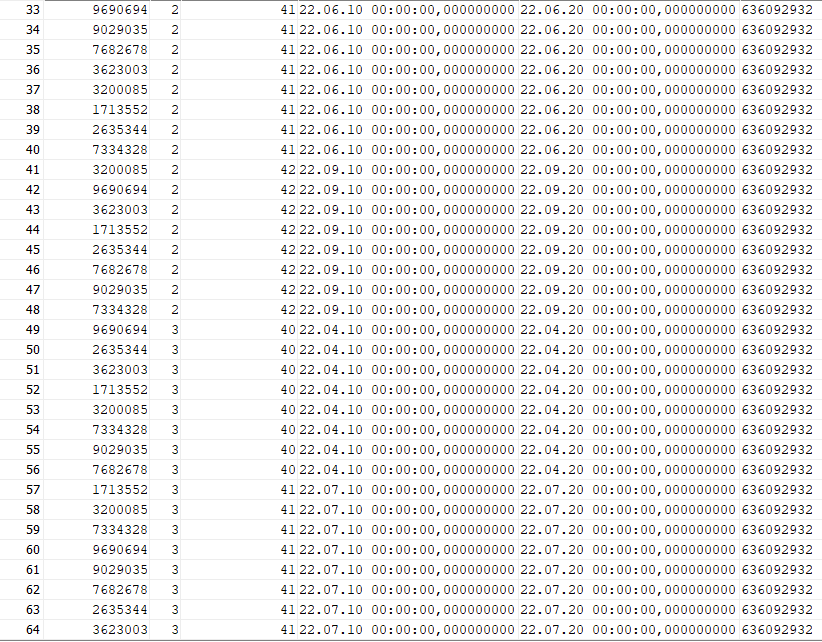


Figura 12 - Containers e stages do navio “636092932”

# US406

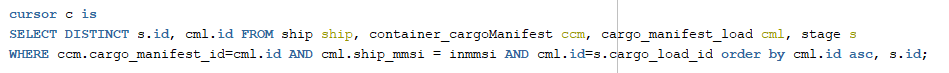
 Nesta US, era necessário descobrir a taxa de ocupação média de todas as *stages* de um navio e mostrar os detalhes das mesmas, mas apenas para as viagens em que a taxa de ocupação média fosse menor a um certo limite. Foi utilizado o mesmo cursor da US405 pois novamente esta *User Story* trabalhava com *stages*.

Figura 13 - Cursor

 Um dos critérios desta US mencionava que apenas viagens já feitas eram consideradas, logo foi colocada a data do dia atual na variável *timestamp* “dataAtual”.

Figura 14 - Data atual

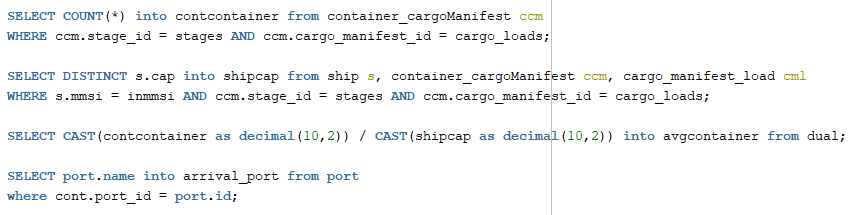
 Foi reutilizado o código da US405, que determinava a média de ocupação de cada *stage*. Essa média era colocada na variável *float* “avgcontainer”.

Figura 15 - Média de ocupação da stage

Depois era certificado se a ocupação da viagem era inferior ao limite, e se a *departure date* e a *arrival date* eram menores à data atual, ou seja, era verificado se a viagem não estava a decorrer no momento ou se ainda ia decorrer. Caso o *id* da *stage* fosse 1 (cont.id), o porto do qual a viagem iniciou a viagem foi no porto que se encontra na tabela *cargo\_manifest\_load*, daí a necessidade da verificação.

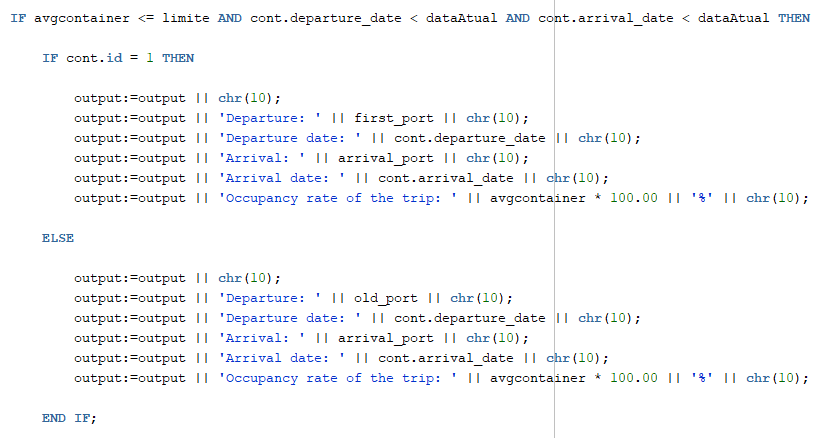


Figura 16 - Verificação do porto e output

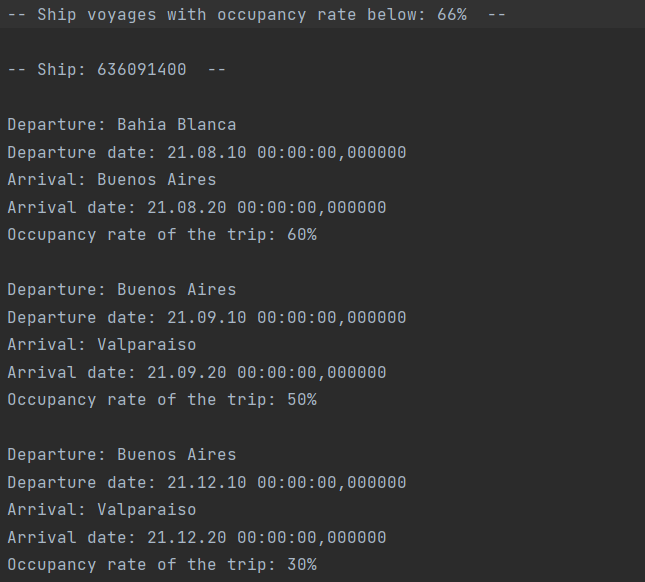
Na figura 17 está demonstrado o *output* final, que são os detalhes de todas as viagens com taxa de ocupação média menor a 66% e na figura 18 estão demonstrados os dados que comprovam o *output* demonstrado na figura 17 estão corretos.

Figura 17 - Output final

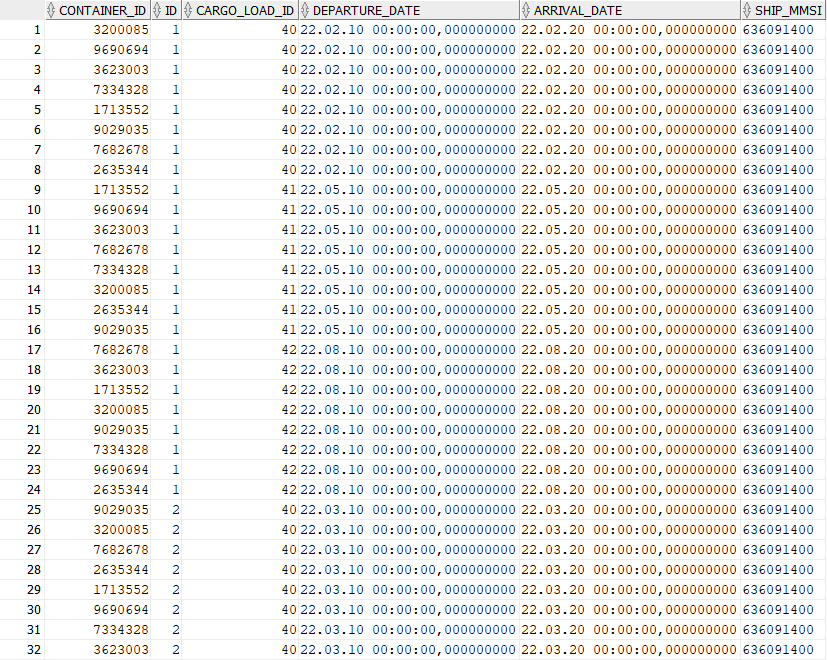
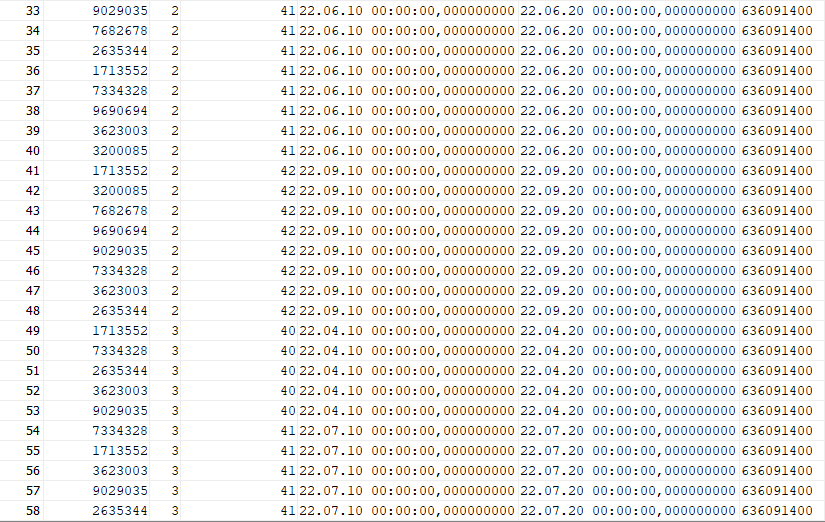


Figura 18 - Containers e stages do navio “636091400" (capacidade do navio: 10)

# US407

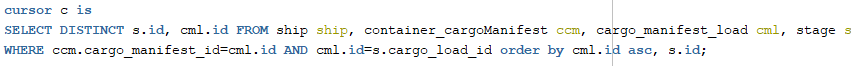
Concluindo, a US407 tem como objetivo criar um mapa de planos de *unload* e *load* de um porto baseado nos *manifests* dos navios e dos camiões e as suas futuras viagens. Utilizei o cursor da US405 pois novamente esta *User Story* exigiu bastante trabalho com *Stages* e *Manifests*.

Figura 19 - Cursor

Para obter a Segunda-Feira da semana seguinte, utilizei o *SELECT* da figura 20, e coloquei a data na variável *timestamp* “nextMonday”.



Figura 20 - Select da Segunda-Feira da semana seguinte

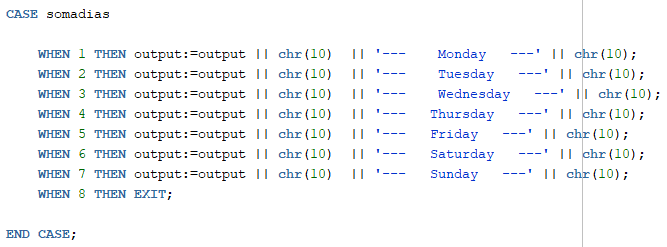
 Decidi mostrar a informação por cada dia da semana, ou seja, era mostrado o mapa de *unload* e *load* da Segunda-Feira da próxima semana, e após toda a informação ser mostrada, passava para o mapa do dia seguinte, ou seja, Terça-Feira, continuando até terminar no Domingo. Para isso, no final de ter feito o mapa de cada dia, somava um dia ao “nextMonday” e somava 1 à variável *integer* “somadias”, que detetava em que dia estava no ciclo e colocava no *output* a partir do *Case* demonstrado na figura 21.

Figura 21 - Case "somadias"

Caso fosse uma carga para dar *unload*, esta estaria em uma *stage* em que a *arrival date* seria o dia a ser da semana a ser analisado, e caso fosse uma carga para dar *load*, esta estaria em uma *stage* na qual a *departure date* seria o dia da semana a ser analisado. Também era necessário verificar se o *port* da *stage* é o porto inserido, que se encontra na variável “inport”.



Figura 22 - Verificação caso fosse carga de unload



Figura 23 - Verificação caso fosse carga de load

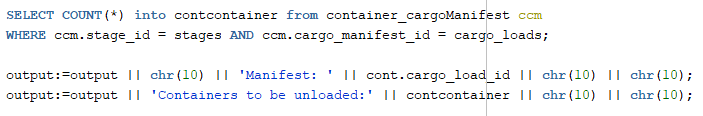
Nos 2 casos (*unload* e *load*) contei os *containers* da *stage* para verificar quantos *containers* iam ser carregados ou descarregados e adicionar ao *output*.

Figura 24 - Contagem de containers

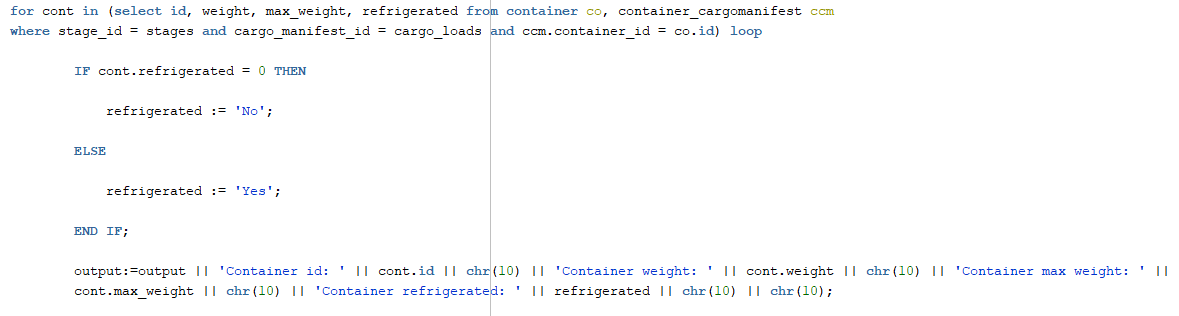
Depois de contar os *containers*, é adicionado ao *output* a informação de todos os *containers* que vão ser carregados ou descarregados. Como o atributo “refrigerated” é *boolean*, caso seja *false*, é adicionado “não” ao output. Se for *true* é adicionado “sim”.

Figura 25 - Informação dos containers

Em seguida, era necessário saber se os containers iriam ser carregados/descarregados em um camião ou em um navio. De modo a descobrir isso, eu colocava o *ship\_mmsi* do *cargo\_manifest\_load* na variável “cargoport”. Se esta fosse *null*, como não existe *ship*, logo a ação vai ser realizada por um *truck*. Nesse caso é inserido o *truck\_id* na variável “truckid” e depois é adicionado ao *output*. Caso esta não seja *null*, quer dizer que o *unload/load* vai ser efetuado por um navio, logo também é necessário mostrar em que posição os *containers* estão no navio para serem descarregados ou a posição em que é para colocar os *containers* no navio.

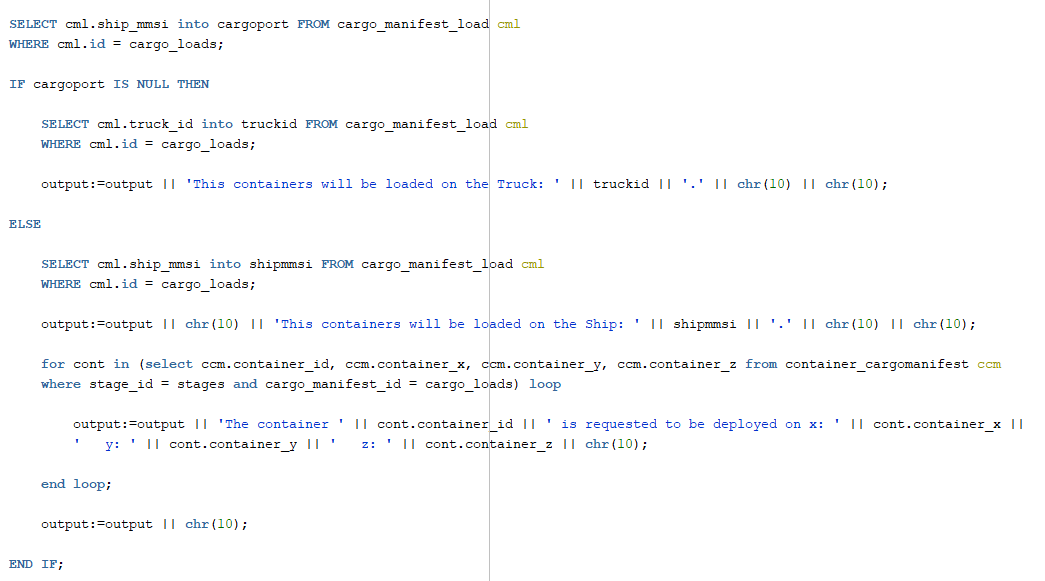


Figura 26 - Truck/ship e posição dos containers no ship

Para terminar, verificamos também a próxima etapa dos *containers*. Para isso é necessário descobrir o *port\_id* ou o *warehouse\_id* da próxima *stage* (cont.id + 1). Caso a variável “stagecount” (que contém a quantidade de *stages* nesse *manifest*) fosse igual à *stage* atual, quer dizer que não existe uma próxima *stage*, ou seja, é o fim da jornada dos *containers*. Só é revelada a próxima etapa dos *containers* caso esteja a ser efetuado um *load*.

Figura 27 - Próxima etapa dos containers



 Se nenhum trabalho fosse registado nesse dia, era adicionado ao *output* “no registered work today”.

Figura 28 - Caso não seja registrado trabalho no dia

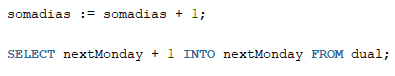
 Após todos os ciclos, era somado o dia à variável *integer* “somadias” e também era adicionado um dia ao “nextMonday”, para ser possível analisar o dia seguinte.

Figura 29 - Soma dos dias

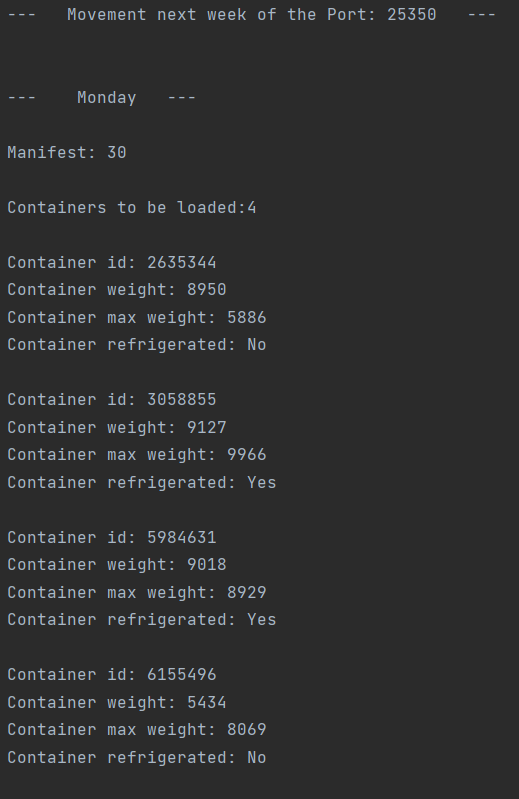
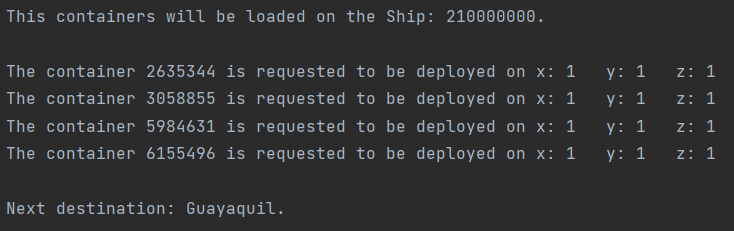
 Na figura 30 está demonstrado uma Segunda-Feira de um *output* final, que são os detalhes de todos os trabalhos que vão ser realizados na Segunda-Feira da semana seguinte.

Figura 30 - Output de uma segunda-feira da semana seguinte