
TourSolver constraints

GEOCONCEPT SAS

Configuring clients	4
General	4
Localisation	7
Optimization constraints	9
Contact information	14
Satisfaction	16
Configuring visits	18
Name (<i>id</i>)	18
Compatibility with the resources (<i>requiredSkills</i>)	18
Require all skills to be compatible (<i>allSkillsRequired</i>)	19
To visit (<i>active</i>)	20
Quantity (<i>quantity[0]</i>)	20
Quantity 2, Quantity 3, ..., Quantity 24 (<i>quantity[1],...quantity[23]</i>)	21
Loading/unloading duration per unit (<i>unloadingDurationPerUnit</i>)	21
Type (<i>type</i>)	22
Visit duration (<i>fixedVisitDuration</i>)	23
Opening time 1, Closing time 1 (<i>timeWindow[0].beginTime,timeWindow[0].endTime</i>)	23
Opening time 2, Closing time 2 (<i>timeWindow[1].beginTime,timeWindow[1].endTime</i>)	24
Opening time 3, Closing time 3 (<i>timeWindow[2].beginTime,timeWindow[2].endTime</i>)	25
Opening time 4, Closing time 4 (<i>timeWindow[3].beginTime,timeWindow[3].endTime</i>)	25
Opening days 1 (<i>possibleVisitDays[0]</i>)	25
Opening days 2 (<i>possibleVisitDays[1]</i>)	26
Opening days 3 (<i>possibleVisitDays[2]</i>)	27
Opening days 4 (<i>possibleVisitDays[3]</i>)	27
End visit before closing time (<i>wholeVisitInTimeWindow</i>)	27
Lateness tolerance (<i>punctuality</i>)	28
Penalty per late hour (<i>delayPenaltyPerHour</i>)	28
Visit frequency (<i>frequency</i>)	29
Last visit date (<i>tsOrderLastVisit</i>)	30
Minimum spacing between visits (<i>tsOrderMinimumSpacing</i>)	30
Duration of a portion of a cut visit (<i>minDuration</i>)	31
Threshold duration for cutting visits (<i>minPartDuration</i>)	31
Exclude resources (<i>excludeResources</i>)	32
Assign resources (<i>assignResources</i>)	32
Outsourcing cost (<i>courierPenalty</i>)	32
Evaluation mode - Name of resource (<i>evaluationInfos.orderOriginalResourceId</i>)	33
Evaluation mode - Day of visit (<i>evaluationInfos.orderOriginalVisitDay</i>)	33
Evaluation mode - Order number (<i>evaluationInfos.orderPosition</i>)	33
Other data (<i>customDataMap</i>)	33
Configuring resources	34
Name (<i>id</i>)	34
Name of departure and arrival points	34
Work time (<i>workStartTime,workEndTime</i>)	34
Lunch break	35

Overtime	36
Hourly cost (<i>workPenalty</i>)	37
Cost per km (<i>travelPenalty</i>)	37
Speed weighting (<i>speedAdjustment</i>)	38
Vehicle capacity (<i>capacity[0]</i>)	38
Vehicle	39
Planning	40
Tour	44
Loading/Unloading	47
Costs	54
Nights	58
Driving legislation	60
Configuring sites	63
Site name (<i>id</i>)	63
Opening times 1 (<i>timeWindow[0].beginTime,timeWindow[0].endTime</i>)	63
Opening times 2 (<i>timeWindow[1].beginTime,timeWindow[1].endTime</i>)	64
Opening times 3 (<i>timeWindow[2].beginTime,timeWindow[2].endTime</i>)	64
Opening times 4 (<i>timeWindow[3].beginTime,timeWindow[3].endTime</i>)	65
Opening days 1 (<i>openingDays[0]</i>)	65
Opening days 2 (<i>openingDays[1]</i>)	65
Opening days 3 (<i>openingDays[2]</i>)	66
Opening days 4 (<i>openingDays[3]</i>)	66
Site management	66
Stock	67

Configuring clients

- ❗ In this documentation, keywords in the default terminology are used as defined in the *My activity* page of the interface.
So, although we will use the word "client" here, the names of the constraints may well vary as a function of the terminology chosen when configuring the application.

General

Code client

This value allows you to identify the customer. This identifier is generated automatically by TourSolver and cannot be edited.

Format: character

Example:

1581948335603-+.124PdenmmwyADe6GVE

It cannot be imported.

Reference

This value allows you to identify the client. The name may be a patronym, a number, or an identifier...

Format: character

Example:

BL001

Assigning a value to this constraint is mandatory.

It can be imported.

Type

This value segments your portfolio database.

For example, it allows you to separate customers into Individuals / Companies, or to categorize the customer base by activity or turnover.

This value has no effect on the optimization but can be useful to set up filters/sorts to run a more targeted visit campaign.

Format: character

Example:

Opportunity

It can be imported.

Designation

The full name of the company or individual to service.

Format: character

Example:

Geoconcept

It can be imported.

Description

This value lets you add comments. Useful when the customer is an opportunity uploaded from the mobile app. Information such as customer availability, access issues, etc. can be rapidly uploaded to the customer description, and can be edited subsequently.

Format: character

Example:

Geographic software editor

It can be imported.

Client

This value complements the Type value.

It will inform as to whether this is an opportunity, or a customer for whom a service commitment is already in place. For users of the mobile app, opportunities uploaded from the field are not considered as customers (their value is set to "0").

If this value is not imported then the default value will be "1".

Format: boolean

Example:

0

It can be imported.

Sector

Free name or default identifier as defined for the sectorization.

Format: character

Example:

Paris territory

It can be imported.

Color

Stores the RGB / CSS customer color code.

Format: character

Example:

C97373

It can be imported.

Creation time

This value identifies the time of creation (via import, or manually) of the customer on the system. This identifier is generated automatically by TourSolver and cannot be edited.

Format: date and time

Example:

19/12/2019 09:08:31

It cannot be imported.

Update date

This value identifies the time the customer was updated on the system (via import or manually). This identifier is generated automatically by TourSolver, and cannot be edited.

Format: date and time

Example:

19/12/2019 09:08:31

It cannot be imported.

Last update

This value identifies the user that updated the customer on the system (via an import or manually): it is generated automatically by TourSolver and cannot be edited.

Format: character

Example:

Admin

It cannot be imported.

Localisation

Address

This is the number, type, and name of the street. This field should be filled if you do not have any geographic coordinates.

Format: character

Example:

162 avenue albert petit

This is not mandatory, but is strongly recommended.

It can be imported.

Additional address information

This field allows you to enter additional address information, which is useful during the final phase of delivery, enabling users of the mobile app to find their destination easily.

Format: character

It can be imported.

City

This is the only address field that must be filled. We strongly recommend filling as many address fields as possible to ensure maximum precision with regard to the address.

Format: character

Example:

Bagneux

Assigning a value to this constraint is mandatory.

It can be imported.

Postal code

This is the post code.

Format: character

Example:

92310

It can be imported.

State

Format: character

It can be imported.

Country

Format: character

Example:

France

It can be imported.

Longitude

If you have WGS84 type geographic coordinates, you can enter them here.

Format: real number

Example:

2.9976290920

It can be imported.

Latitude

If you have WGS84 type geographic coordinates, you can enter them here.

Format: real number

Example:

48.988000772

It can be imported.

Geocoding type

This field can only be edited by TourSolver.

Format: character

Example:

City

It cannot be imported.

Optimization constraints

Organization

This value segments the customer database according to how it is structured, this being allied in turn to user rights allocated. For example, a resource and/or a planner affiliated to "Paris", will not see customers classified as "Bordeaux".

Format: character

Example:

PARIS area

It can be imported.

Last visit date

This constraint allows you to take the date of the last visit preceding the route to be optimized into account, and so allows the application to calculate and respect as far as possible a time delay between visits.

Format: date

Example:

19/01/2019

It can be imported.

Next visit date

This value permits information about the next visit to be shared, if this information has been given out by the planner following a planning session.

Format: date

Example:

30/03/2019

It cannot be imported.

Visit duration

This is the least possible time that can be spent on the visit, that is to say a minimum customer visit time.

Format: HH:MM:SS

Example:

00:15:05

It can be imported.

Possible visit days 1

This constraint contains all possible *Opening days* for visits (up to 64) for which the time window as defined by the Opening time 1 and Closing time 1 constraints applies.

Up to 4 opening days can be included.

Format: numeric or date

Example:

1-5

It can be imported.

Possible visit days 2

This constraint contains all possible *Opening days* for visits (up to 64) for which the time window as defined by the Opening time 2 and Closing time 2 constraints applies.

Up to 4 opening days can be included.

Format: numeric or date

Example:

1-5

It can be imported.

Possible visit days 3

This constraint contains all possible *Opening days* for visits (up to 64) for which the time window as defined by the Opening time 3 and Closing time 3 constraints applies.

Up to 4 opening days can be included.

Format: numeric or date

Example:

1-5

It can be imported.

Possible visit days 4

This constraint contains all possible *Opening days* for visits (up to 64) for which the time window as defined by the Opening time 4 and Closing time 4 constraints applies.

Up to 4 opening days can be included.

Format: numeric or date

Example:

1-5

It can be imported.

Opening time 1 and Closing time 1

These constraints allow you to define a time window over which the visit may be performed. Opening time 1 corresponds to the time from which the visit can be performed. Closing time 1 corresponds to the time after which the visit can no longer be performed. This time window will apply to all days unless Opening days 1 has been defined. In this case, the time window is applied only to this value.

If a timeslot has not been entered for a given visit, the visit can be fulfilled at any time, on any of the Opening days defined, and within the limits set by the resource's working hours. You can define up to 4 different time slots (and the associated days) for each visit.

Format: HH:MM:SS

Example Opening time 1:

09:50:00

Example Closing time 1:

10:50:00

They can be imported.

Opening time 2 and Closing time 2

These constraints allow you to define a time window over which the visit may be performed. Opening time 2 corresponds to the time from which the visit can be performed. Closing time 2 corresponds to the time after which the visit can no longer be performed. This time window will apply to all days unless Opening days 2 has been defined. In this case, the time window is applied only to this value.

If a timeslot has not been entered for a given visit, the visit can be fulfilled at any time, on any of the Opening days defined, and within the limits set by the resource's working hours. You can define up to 4 different time slots (and the associated days) for each visit.

Format: HH:MM:SS

Example Opening time 2:

09:50:00

Example Closing time 2:

10:50:00

They can be imported.

Opening time 3 and Closing time 3

These constraints allow you to define a time window over which the visit may be performed. Opening time 3 corresponds to the time from which the visit can be performed. Closing time 3 corresponds to the time after which the visit can no longer be performed. This time window will apply to all days unless Opening days 3 has been defined. In this case, the time window is applied only to this value.

If a timeslot has not been entered for a given visit, the visit can be fulfilled at any time, on any of the Opening days defined, and within the limits set by the resource's working hours. You can define up to 4 different time slots (and the associated days) for each visit.

Format: HH:MM:SS

Example Opening time 3:

09:50:00

Example Closing time 3:

10:50:00

They can be imported.

Opening time 4 and Closing time 4

These constraints allow you to define a time window over which the visit may be performed. Opening time 4 corresponds to the time from which the visit can be performed. Closing time 4 corresponds to the time after which the visit can no longer be performed. This time window will apply to all days unless Opening days 4 has been defined. In this case, the time window is applied only to this value.

If a timeslot has not been entered for a given visit, the visit can be fulfilled at any time, on any of the Opening days defined, and within the limits set by the resource's working hours. You can define up to 4 different time slots (and the associated days) for each visit.

Format: HH:MM:SS

Example Opening time 4:

09:50:00

Example Closing time 4:

10:50:00

They can be imported.

Required skills

This constraint designates the criteria to be fulfilled as stipulated by the customer, that the resource will need to respect to be able to visit them.

Format: character - list of words separated by commas (maximum of 64 different words for all files of visits and resources handled)-

Example:

robert,ana

It can be imported.

Frequency

This constraint allows you to specify that a visit must be performed several times over a period of several days. TourSolver attempts to optimize by balancing as best it can the gaps between visits while also taking the other constraints into account. The frequency does not guarantee absolute respect for the specified periods of time. An indicator showing non-respect for frequency periods in the TourSolver interface informs you of the number of visits that do not fall within the optimum visiting period.

Format: character

Example:

1/5

It can be imported.

Minimum spacing between visits

This constraint allows you to indicate the minimum number of days required between two visits of a same customer.

Format: integer

Example:

5

It can be imported.

Maximum spacing between visits

This constraint allows you to indicate the maximum number of days required between two visits of a same customer.

Format: integer

Example:

5

It can be imported.

Frequency type

This is a descriptive value, designed to enable simpler filtering of data in the interface.

Format: character

Example:

Weekly

It can be imported.

Contact information

First name

This value allows you to enter the first name of a contact.

Format: character

Example:

Alexis

It can be imported.

Last name

This value allows you to enter the last name of a contact.

Format: character

Example:

BERGER

It can be imported.

Occupation

This value allows you to enter the staff position for a contact.

Format: character

Example:

Product Owner

It can be imported.

Email address

This value allows you to enter the email address for a contact.

This information will be used for sending ""Customer feedback"" notifications if this function is enabled in TourSolver and if the Receive notifications field for this customer is active.

Format: character

Example:

alexis.berger@geoconcept.com [mailto:alexis.berger@geoconcept.com]

It can be imported.

Get notifications

This value allows you to authorize the sending of an SMS or email to a contact.
By default the value is set to 0.

Format: boolean

Example:

0

It can be imported.

Landline phone

This value is additional information and is not used in TourSolver.

Format: character

Example:

0172749887

It can be imported.

Mobile phone

This value allows you to send SMS messages to a contact.

This information will be used in the sending of ""Customer feedback"" notifications if this function is enabled in TourSolver and if the Receive notifications field for the customer is active.

Format: character

Example:

0639876719

It can be imported.

Satisfaction

Last feedback

This value contains the score for the recipient (from 1 to 5). It cannot be edited.

Format: integer

Example:

5

It cannot be imported.

Last comment

This value contains the customer's comments when feedback is received from ""customer feedback"" exchanges.

Format: character

Example:

The delivery staff were very helpful!

It cannot be imported.

Last visit id

This value contains the identifier for the last visit made to this customer.

Format: integer

Example:

BL00981

It cannot be imported.

Photo URL

This value lets you store photographs taken in the field by users of the mobile application.

Format: URL (list)

Example:

<http://geoconcept/photo13>, <http://geoconcept/photo21>

It cannot be imported.

Configuring visits

- In this documentation, keywords in the default terminology are used as defined in the *My activity* page of the interface.

So, although we will use the word "visit" here, the names of the constraints may well vary as a function of the terminology chosen when configuring the application.

- After each constraint name, the brackets contain the equivalent name assigned to this constraint under [TourSolver Cloud API](https://geoservices.geoconcept.com/ToursolverCloud/api-book.html) [https://geoservices.geoconcept.com/ToursolverCloud/api-book.html].

Name (*id*)

This value is the visit identifier. The name may be a patronym, a number, an identifier... It is mandatory.

Format: character

Example:

Mr Dupont / BL0012

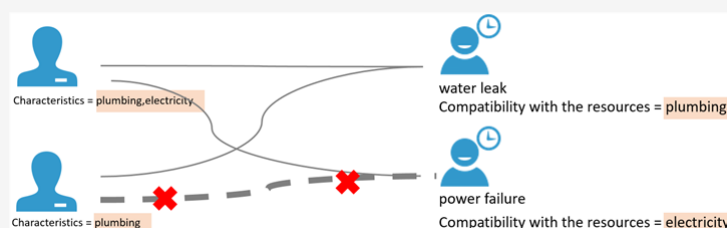
Compatibility with the resources (*requiredSkills*)

This constraint designates criteria requested by the visit that need to be fulfilled to enable the resource to perform the visit.

Format: list of words separated by commas (maximum of 64 different words for all visit and resource files handled)

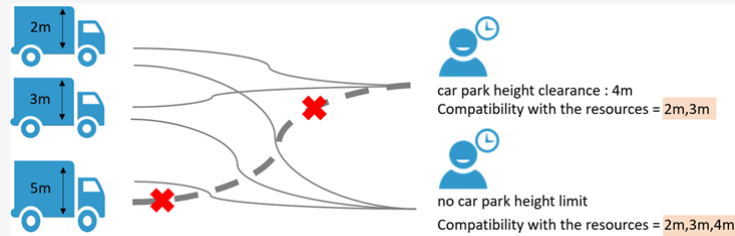
Example 1:

For a customer needing a maintenance intervention relating to plumbing, simply indicate "plumbing". At least one of the resources must qualify with respect to this specific competence to enable the customer visit to go ahead.



Example 2:

For a customer with limited vehicle height access, for example 4m maximum, you can indicate "2m, 3m" since the delivery will require a vehicle of less than 4m. It will be one of the resources with a specific height of 2m or 3m that will be able to visit.



Require all skills to be compatible (*allSkillsRequired*)

If the Compatibility with the resources constraint is used (and the values are numerous), this constraint allows you to indicate whether only one of the characteristics in the list is required or if the full list of characteristics is required.

Format: binary

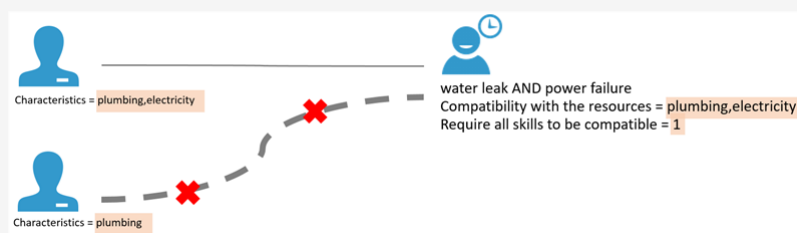
If the value = 0, just one skill is required.

If the value = 1, the full list of skills is required.

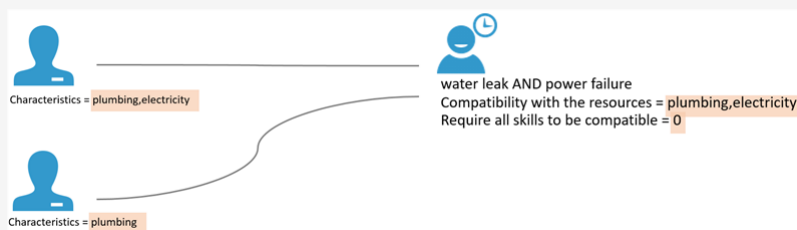
The default value is equal to 1.

Example 1:

For a customer needing a maintenance intervention related to plumbing AND electricity, indicate "plumbing, electricity". At least one of the resources must then fulfill these 2 skills for the customer to be visited. In this case you should put a value of 1 under the Require all skills to be compatible constraint.



If the Require all skills to be compatible constraint = 0 then both these skills will not have to be fulfilled.



To visit (*active*)

This constraint allows you to take into consideration visits to be performed without having to add or subtract any data in the full data set.

Format: binary

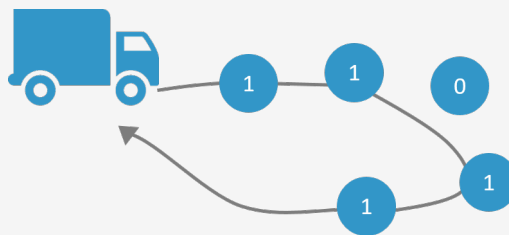
If the value = 0, the visit is not included in the route.

If the value = 1, the visit will be performed.

The default value is equal to 1.

Example:

For 5 visits, only 1 will not be included in this route.



Quantity (*quantity[0]*)

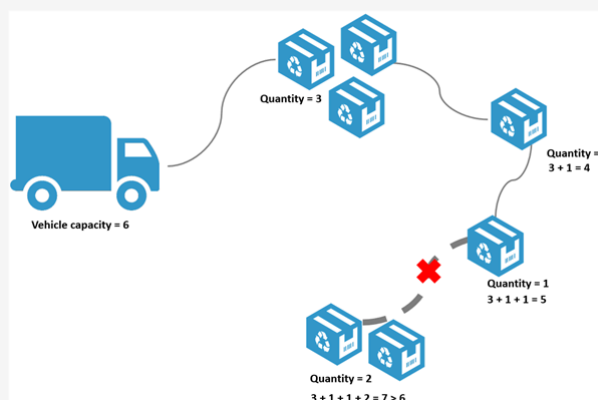
This value represents the quantity to deliver or collect. The user sets the units for the quantity (kilos, m3, litres...) and ensures coherence with the Vehicle capacity constraint.

Format: number

Possible values for this constraint range from 0 to 2147483. TourSolver does not recognise more than 3 significant figures after the comma.

Example:

In the case of a parcel delivery, indicate the number of parcels to deliver (3 for 3 parcels), in the case of liquid goods, indicate the number of litres (3000 for 3000 litres), and in the case of heavy duty packaging, it is the volume that is indicated (2 for 2m3).



- ! A visit can only be serviced by a single, unique resource. A visit cannot be fulfilled by several resources, the combined respective capacities of whom might satisfy the visit criteria. So this means that a visit, for which a Quantity constraint exceeds the corresponding maximum Vehicle capacity constraint as defined for resources is considered to be impossible and is not taken into account in the resolution of the problem. To work around this problem, the user may break the visit down into several visits.

Quantity 2, Quantity 3, ..., Quantity 24 (*quantity[1],...quantity[23]*)

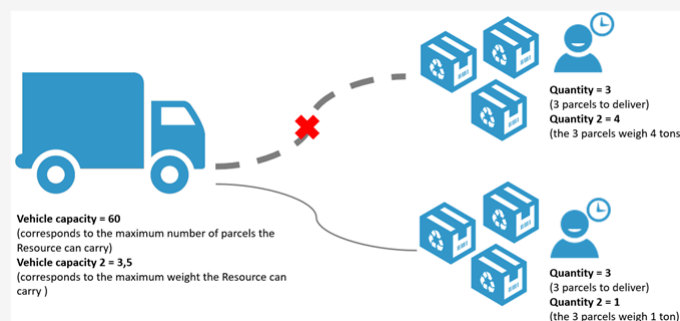
In addition to the Quantity parameter, TourSolver suggests 24 measurement parameters for goods to be delivered or collected. Each of them needs to be put into correspondence with the Vehicle capacity to Vehicle capacity 24 parameters for resources and enables different situations to be taken into account.

Format: number

Possible values for this constraint range from 0 to 2147483. TourSolver does not recognise more than 3 significant figures after the comma.

Example:

Deliveries or collections of heavy and voluminous objects. We use Quantity and Vehicle capacity for the weight, Quantity 2 and Vehicle capacity 2 for the volume. The loading of resources and the delivery or collection from customers will be optimised as a function of the double constraint of weight and volume.



Loading/unloading duration per unit (*unloadingDurationPerUnit*)

This value corresponds to the time needed to deliver a unit of merchandise. This value is multiplied by the value entered under the Quantity constraint. This constraint applies only to the Quantity constraint of the visit, and not to the 23 other possible quantities.

Format: HH:MM:SS or in seconds (20 for 20 seconds)

Example:

in the case of a delivery of 10 parcels to a customer: the estimated unloading time for a single parcel is 3 minutes. Enter the value 00:03:00 (or 180 seconds) for the Loading/unloading duration per unit and a

value of 10 in the Quantity constraint, and this will mean the resource will enter 30 minutes for the delivery (3min x 10 parcels).

Type (*type*)

This constraint allows you to indicate if the visit concerns a delivery/drop-off or a return/collection/pick-up.

Format: binary

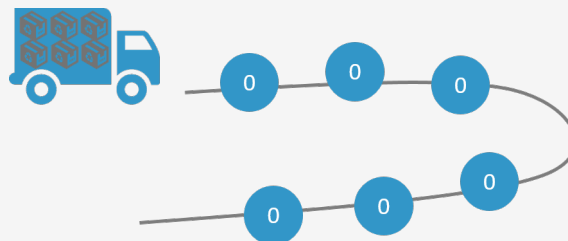
If the value = 0, the route is in delivery mode;

If the value = 1, the route is in collect mode;

If the value = 2, the route is in delivery mode to be executed before any collection can take place.

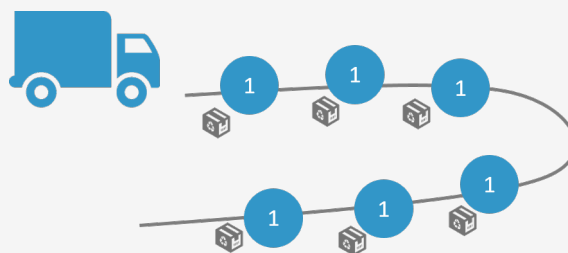
Example 1:

a delivery route (Type = 0)



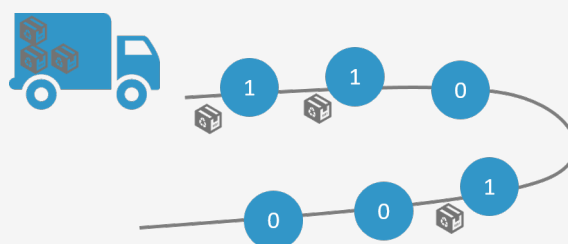
Example 2:

a collection route (Type = 1)



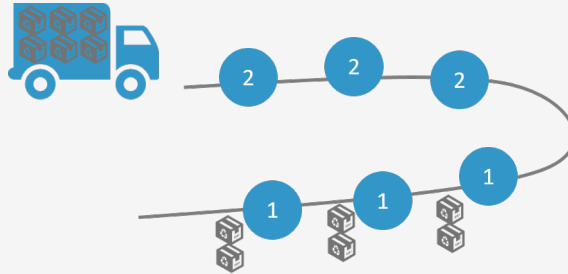
Example 3:

a delivery route (Type = 0 / collection (Type = 1)



Example 4:

a sequenced delivery route (Type = 2) / collection (Type = 1)



For a single visit, one delivery and one collection can be performed. In this case you must duplicate the visit line, modify the type, and if need be the quantity (to deliver or collect).

Name	Type	Quantity
Client1	0	20
Client1	1	10

Visit duration (*fixedVisitDuration*)

This is the least possible time that can be spent on the visit, that is to say a minimum customer visit time. To this time we can add the time entered under the Loading/unloading duration per unit constraint.

Format: HH:MM:SS

Example:

for a delivery, the Visit duration may correspond to the parking time, the time it takes to get the delivery note signed, to manoeuvre the vehicle's tailgate, etc, to add to the time needed for the unloading operation as specified in the Loading/unloading duration per unit constraint.

Opening time 1, Closing time 1

(*timeWindow[0].beginTime,timeWindow[0].endTime*)

These constraints allow you to define a time window over which the visit may be performed. Opening time 1 corresponds to the time from which the visit can be performed. Closing time 1 corresponds to the time after which the visit can no longer be performed. This time window will apply to all days unless Opening days 1 has been defined. In this case, the time window is applied only to this value.

If a time window is not documented for any given visit, the visit can be performed at any time, on any of the Opening days as defined and within the resource's working time limits. You can define up to 4 different time windows (and their associated days) for each visit.

Format: HH:MM:SS

Example 1:

Opening time 1 = 07:30:00 and Closing time 1 = 11:30:00 define a time window of 4 hours during which the resource can visit the customer.

Example 2:

The customer must receive delivery between 11:00pm and 03:00am. In this case, you need to shift the times (in one direction or the other) so that they are contained in one day, and retrieve the sequencing of the visits to be performed.

Opening time 1 = 19:00:00 and Closing time 1 = 23:00:00 define a time window of 4 hours during which the resource can visit the customer.

After the optimisation has been performed, you will need to shift the series of visiting times again so they fall within the original time window.

This customer has been scheduled for a visit at 09:00pm by TourSolver, however in reality, they will be visited at 01:00am (=23:00:00 + 2 hours).

- ❗ Windows overlapping several days are not handled: for example, the user cannot use the window that starts at 23:00:00 and finishes at 03:00:00 the following morning without shifting the existing time assignments.

- 💡 A visit cannot start before the start time. By default, a visit that starts in the time window can continue beyond the end time. If you want a visit to finish before Closing time 1 you will need to activate End visit before closing time.

Opening time 2, Closing time 2

(timeWindow[1].beginTime,timeWindow[1].endTime)

These constraints allow you to define a second time slot that will apply to visit days defined under Possible visit days 2. Opening time 2 corresponds to the time from which the visit can be performed. Closing time 2 corresponds to the time beyond which the visit can no longer be performed.

Format: HH:MM:SS

Example 1:

The customer can receive delivery in the morning between 09:00am and 12:00pm and in the afternoon between 02:00pm and 05:00pm. For this, we will use the constraint:

Opening time 1 = 09:00:00;

Closing time 1 = 12:00:00;

Opening time 2 = 14:00:00;

Closing time 2 = 17:00:00.

Example 2:

The customer can receive delivery on Monday between 07:00am and 05:00pm and on Wednesday between 09:00am and 06:00pm, in this case:

Opening time 1 = 07:00:00;

Closing time 1 = 17:00:00;

Opening days 1 = 1;

Opening time 2 = 09:00:00;

Closing time 2 = 18:00:00;

Opening days 2 = 3.

Opening time 3, Closing time 3 (*timeWindow[2].beginTime,timeWindow[2].endTime*)

These constraints allow you to define a second time slot that will apply to visit days defined under Possible visit days 3. Opening time 3 corresponds to the time from which the visit can be performed. Closing time 3 corresponds to the time beyond which the visit can no longer be performed.

Format: HH:MM:SS

Opening time 4, Closing time 4 (*timeWindow[3].beginTime,timeWindow[3].endTime*)

These constraints allow you to define a second time slot that will apply to visit days defined under Possible visit days 4. Opening time 4 corresponds to the time from which the visit can be performed. Closing time 4 corresponds to the time beyond which the visit can no longer be performed.

Format: HH:MM:SS

Opening days 1 (*possibleVisitDays[0]*)

This constraint contains all possible visit days (up to 64) to which the time window defined by the Opening time 1 and Closing time 1 constraints applies.

Format: numeric or date


These days may be defined one by one (1,4) or by time slot (1-4), as an integer (1,2, ...64) or in date format (14/05/2016 => 17/07/2016).

Example 1:

Opening days 1 = 1,2,5 (or 14/05/2016,15/05/2016,18/05/2016) signifies that the visit can take place on days 1,2 and 5 in the planning (or the 14/05/2016, 15/05/2016 and 18/05/2016) at the times defined by the Opening time 1 and Closing time 1 constraints.

Example 2:


Opening days 1 = 1-5 (or 14/05/2016 => 18/05/2016) signifies that the visit may take place on all days 1, 2, 3, 4 and 5 in the planning (or between the 14/05/2016 and the 18/05/2016) at the times defined by the Opening time 1 and Closing time 1 constraints.

 If you want to indicate that a visit must be performed on day 2 AND on day 4 of the week, you will need to create two Visit lines indicating 2 on Opening days 1 on the first line and 4 on the second line...

Customers	Possibles Days 1
Customer A	2
Customer A	4

...or use the Visit frequency constraint.

Customers	Possibles Days 1	Frequency
Customer A	2,4	2/5

 Caution: this constraint must be put into correspondence with the Worked days constraint under resources and use the same format (date or number). None of the dates for Opening days 1 can fall before the oldest date for Worked days.

Opening days 2 (*possibleVisitDays[1]*)

This constraint contains all Opening days (up to 64) to which the time window defined by the Opening time 2 and Closing time 2 constraints applies. These days may be defined one by one, or by time window, and follow the same syntax as Opening days 1.

Example 1:

The customer can receive delivery on Monday between 07:00am and 05:00pm and on Wednesday between 09:00am and 06:00pm:

Opening time 1 = 07:00:00, Closing time 1 = 17:00:00, Opening days 1 = 1 and Opening time 2 = 09:00:00, Closing time 2 = 18:00:00, Opening days 2 = 3

Example 2:

The customer can receive delivery from Monday to Friday between 07:00am and 05:00pm and on Saturday between 09:00am and 12:00pm:

Opening time 1 = 07:00:00, Closing time 1 = 17:00:00, Opening days 1 = 1-5 and Opening time 2 = 09:00:00, Closing time 2 = 12:00:00, Opening days 2 = 6

Opening days 3 (*possibleVisitDays[2]*)

This constraint contains all Opening days (up to 64) to which the time window defined by the Opening time 3 and Closing time 3 constraints applies. These days may be defined one by one, or by time window, and follow the same syntax as Opening days 1.

Opening days 4 (*possibleVisitDays[3]*)

This constraint contains all Opening days (up to 64) to which the time window defined by the Opening time 4 and Closing time 4 constraints applies. These days may be defined one by one, or by time window, and follow the same syntax as Opening days 1.

End visit before closing time (*wholeVisitInTimeWindow*)

This constraint allows you to define whether the visits can finish after the end of the indicated time window. By default, a visit that starts in the time window can continue beyond the finish time. If it is desirable for the visit to end before Closing time 1, this constraint must be activated.

Format: binary

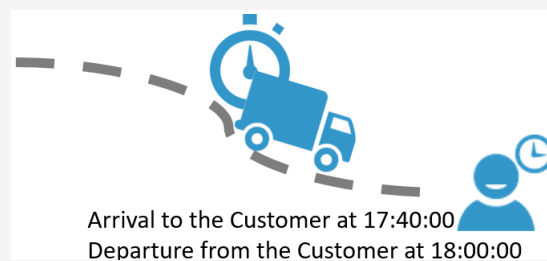
Example:



In the event that End visit before closing time = 0



In the event that End visit before closing time = 1



Lateness tolerance (*punctuality*)

This value allows you to rate the gravity of late delivery. By default, TourSolver will attempt to avoid any late delivery. It will not be useful to assign an intolerance to late delivery for all visits, it will be preferable to work with differences between tolerances.

Format: numeric (value between 1 and 5 inclusive.)

A value for tolerance set to 1 corresponds to a null Penalty per late hour, that is to say that the visit can be performed late. Conversely, a tolerance value set at 2 or more obliges TourSolver to avoid late arrival at the visit, as far as this is possible. Visits having a tolerance value set at 5 will be less at risk of being performed late than visits for whom the tolerance value is set to 2.

Tolerance is respected by a resource when they arrive at the visit within the time window defined, and not if they finish the visit within this time interval (unless the End visit before closing time constraint has been activated).

Example:

This constraint is best described by an image showing a tolerance cursor on a slider.



Penalty per late hour (*delayPenaltyPerHour*)

In effect, this is an additional cost applied per hour of lateness. This constraint affords the user finer control over respect for visiting time windows. Caution: using constraints linked to tolerance can have a blocking effect on the optimisation calculation by making lateness the main *priority*.

Format: numeric (integer or real number)

Example:

I accept making a series of detours and therefore covering extra kilometers in order to better respect the delivery times as defined.

I accept travelling an extra 70 kilometers to avoid 5 minutes late arrival at the customer.

So, $0.5 \times 70 \times 12 (=60/5\text{min}) = 420$

(0.5 being the expression of the resource's Cost per km constraint)

The value to enter under the Penalty per late hour constraint is 420.

Diagram showing a route drawn using the Penalty per late hour constraint

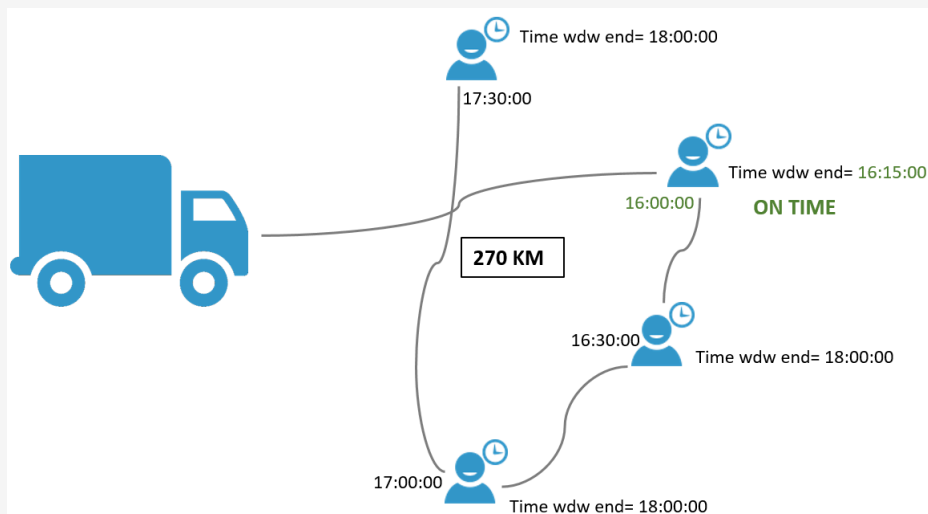
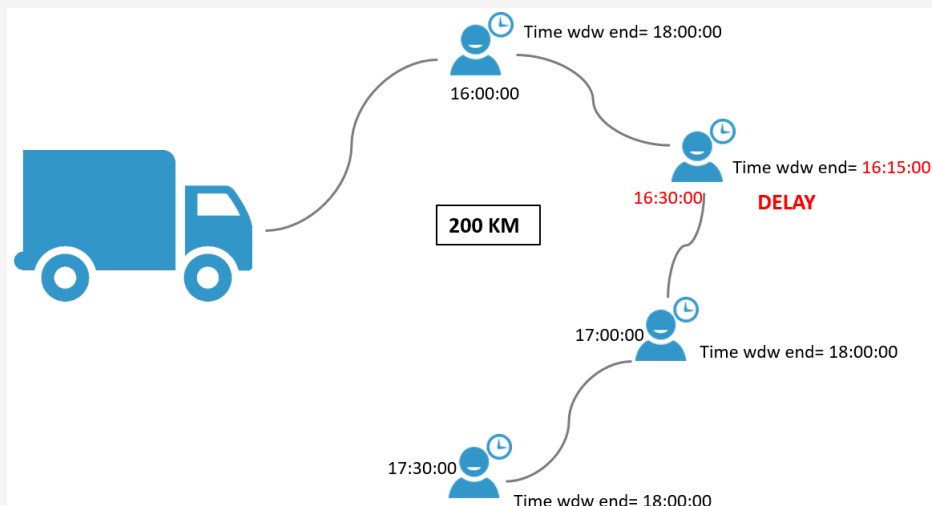


Diagram showing a route drawn without using the Penalty per late hour constraint



Visit frequency (*frequency*)

This constraint allows you to specify that a visit must be performed several times over a period of several days. TourSolver attempts to optimize by balancing as best it can the gaps between visits while also taking the other constraints into account. The frequency does not guarantee absolute respect for the specified periods of time. An indicator showing non-respect for frequency periods in the TourSolver interface informs you of the number of visits that do not fall within the optimum visiting period.

Format: character (1/5)

- ! If this constraint is used, the Worked days constraint defined for resources must have an assigned value.

Example 1:

Visit frequency = 3/5 means that a customer must be visited 3 times in 5 days. TourSolver searches for a solution that respects a gap of 2 days between each visit.

Example 2:

Visit frequency = 3/5* means that a customer must be visited 3 times in 5 days. TourSolver tries to arrange for 3 visits for the same customer to be performed by the same resource. In this case, respecting the gap between 2 visits takes second priority.

Example 3:

Visit frequency = 1/5 means that a customer must be visited 1 times in 5 days. If the constraint Worked days = 1-5, the customer will be visited one times between 1 to 5 days, second times between 6 to 10 days and third times between 11 to 15 days. the constraint Minimum spacing between visits allows you to respect a gap between each visit.

Last visit date (*tsOrderLastVisit*)

This constraint to specify from how many days the last visit was performed before starting the current planning to respect the gap between the visits.

Format: negative integer

! If the number of recurrences of the frequency is more than one, the constraint is not considered.

Example 1:

Visit frequency = 1/5

Minimum spacing between visits = 2

Last visit date = -1

The customer will be visited only from day 2, because if to visit day 1, the gap between visits does not respect the minimum spacing of 2 days.

Minimum spacing between visits (*tsOrderMinimumSpacing*)

This constraint allows you to indicate the minimum number of days required between two visits of a same customer.

Format: integer

! If the number of recurrences of the frequency is more than one, the constraint is not considered.

Example 1:

Visit frequency = 1/5

Minimum spacing between visits = 3

Worked days = 1-15

To respect a gap of 2 days between 2 visits, the customer don't must be visited 5,8,11 days, but 5,9,12 days.

Duration of a portion of a cut visit (*minDuration*)

This constraint corresponds to the minimum duration authorising the splitting of one visit into several visits. The visit can be broken down from one day to the next, or at the time of the lunch break. This value is intimately linked to the Visit duration constraint.

Format: HH:MM:SS

Example:

Let's take a visit A, for which the Visit duration is 4h duration, and a visit B of 30 minutes duration. Duration of a portion of a cut visit is equal to 01:00:00 and so all customers for whom the Visit duration is more than 01:00:00 will be splittable into several visits. Visit A will be splittable if need be, but not visit B. The minimum cutting time for each of these visits is defined by the Threshold duration for cutting visits constraint.

i A visit cannot be split into more than 16 blocks.

Threshold duration for cutting visits (*minPartDuration*)

This value corresponds to the minimum duration below which a visit can no longer be broken down into several visits.

Format: HH:MM:SS

Example:

For a customer for whom the Visit duration is equal to 04:00:00. The Duration of a portion of a cut visit constraint is equal to 01:00:00, which means that the visit can be split. Threshold duration for cutting visits is equal to 01:30:00, which means that following splitting, no visit can last less than 1 hour 30 minutes.

04:00:00

04:00:00

02:00:00

02:00:00

01:00:00

01:30:00

01:30:00

02:30:00

01:30:00

01:00:00

01:00:00

01:00:00

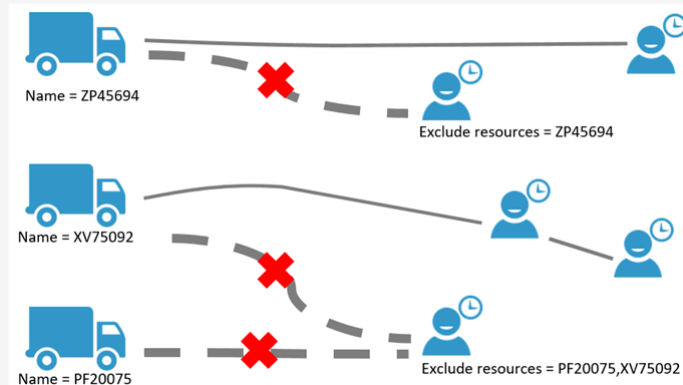
01:00:00

Exclude resources (*excludeResources*)

This constraint allows the user to specify which resources may not perform a visit. The value to enter under this constraint must be the same as the value entered under the Name constraint for resources.

Format: character, in the form of a list of items separated by commas if there are multiple resources.

Example:

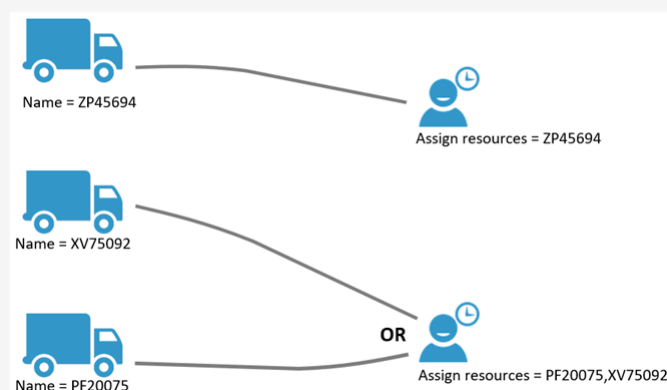


Assign resources (*assignResources*)

This constraint allows the user to specify which resources may perform a visit. The value to enter under this constraint must be the same as the value entered under the Name constraint for each resource.

Format: character, in the form of a list of items separated by commas if there are multiple resources.

Example:



Outsourcing cost (*courierPenalty*)

This constraint allows the user to specify that an order can be delivered by an external courier service. The optimisation engine calculates whether it will be better to send the order using this service, or

whether it is best integrated in a standard delivery trip. These visits do not appear anywhere in the optimisation (routes, planning...). The Outsourcing cost is integrated within the total optimisation cost.

Format: numeric (integer or real number)

Example:

Customer A 1000 and Customer B 100

Evaluation mode - Name of resource (*evaluationInfos.orderOriginalResourceId*)

This value is used in evaluation mode. It allows you to assign a visit to a specific resource. It is especially useful for evaluating existing routes.

Format: character

Evaluation mode - Day of visit (*evaluationInfos.orderOriginalVisitDay*)

This value is used in evaluation mode. During a multi-day route, this constraint allows you to specify a specific route day for a visit. The value of this constraint must be identical to one of the values for the Worked days constraint defined for the specified resource under Evaluation day - Name of resource, and also, where defined, to one of the Opening days for this visit.

Format: character

Evaluation mode - Order number (*evaluationInfos.orderPosition*)

This value is used in evaluation mode. This constraint allows you to specify the visit rank in this route. All visits must have this information filled in in order for the constraint to be applied.

Format: number

Other data (*customDataMap*)

This constraint plays no role whatsoever in the optimisation, it allows you to select other descriptive data on visits, who will be consultable in the results (comments, telephone number, ...). It will suffice to select the columns and they will appear under the different results generated by TourSolver (planning, Roadbooks, etc.) A maximum of 50 columns can be stored.

Configuring resources

- ❗ In this documentation, keywords in the default terminology are used as defined in the *My activity* page of the interface.
So, although we will use the word "resource" here, the names of the constraints may well vary as a function of the terminology chosen when configuring the application.

- ❗ After each constraint name, the brackets contain the equivalent name assigned to this constraint under [TourSolver Cloud API](https://geoservices.geoconcept.com/ToursolverCloud/api-book.html) [https://geoservices.geoconcept.com/ToursolverCloud/api-book.html].

Name (*id*)

This value is the resource identifier. The name may be a patronym, a number, an identifier, a registration number...

Format: character (maximum 128 characters)

Example:

Mr Dupont / ZP65294 / Peugeot607

Assigning a value to this constraint is mandatory.

This value must be unique.

Name of departure and arrival points

This value allows you to identify start and arrival points. The name may be a patronym, a number, an identifier, a town...

Format: character

The default values in the results are: *Departure* and *Return to depot*.

Example:

Departure Bagneux / Grenoble branch

Work time (*workStartTime,workEndTime*)

These constraints allow you to define a time window in which the resource can work. The time at which the resource starts and stops to work.

This time window will be applied to all days, except if Worked days is defined. In this case the time window is applied only to this value.

The user can define up to 4 different time windows (and their associated days) for each resource.

Format: HH:MM

Example 1:

07:30 to 11:30 means that the resource starts to work at 07:30am and returns at 11:30am at the latest.

Example 2:

The resource works between 11:00pm and 03:00am in the morning. In this case, you need to shift all the times recorded (in one direction or the other) so that they are all contained in one day, and retrieve the sequencing of the visits to be performed. 19:00 to 23:00 define a time window of 4 hours during which the resource can work. Following optimisation, you will need to shift all the times recorded for visits again to restore the original time window. If the customer has been planned by TourSolver at 09:00pm, in reality that customer will be visited at 01:00am (=23:00 plus 2 hours).

- ⓘ Time windows that overlap several days are not handled. The user cannot use any time window starting at 23:00 and ending at 03:00 the following day without shifting the times overall.

Lunch break

These constraints allow you to define a lunch break.

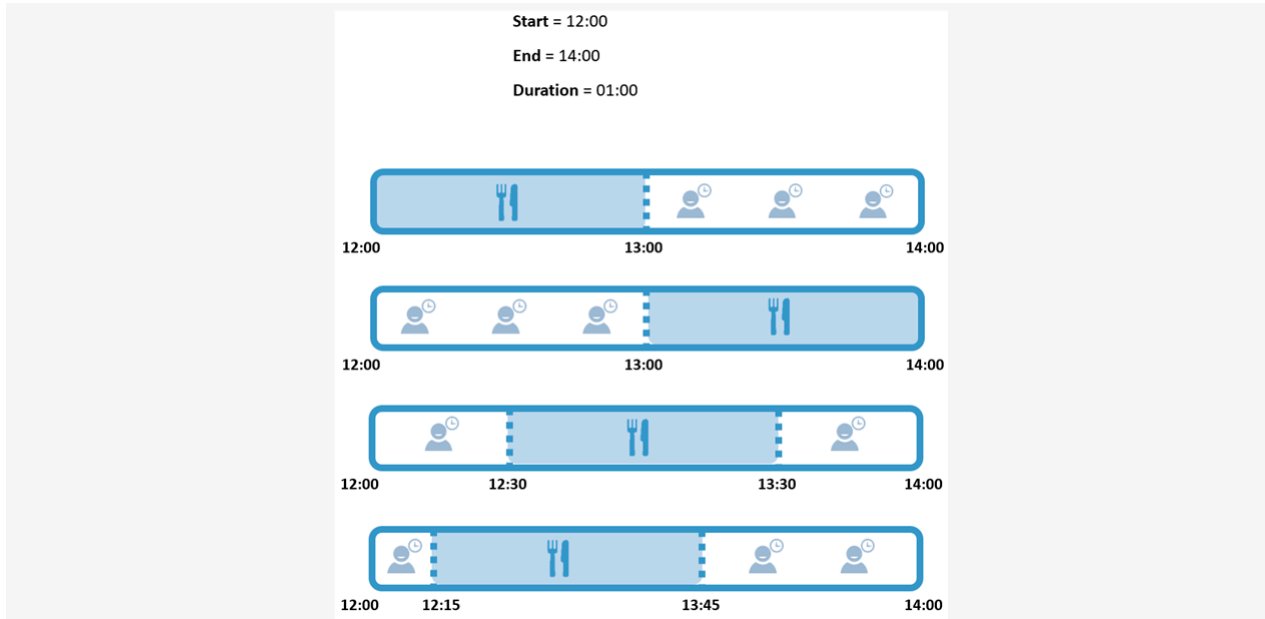
Format: HH:MM

Start (*lunch.start*): This constraint indicates the time from which the resource's lunch break can start.

End (*lunch.end*): This constraint indicates the latest time at which the resource's lunch break may finish. If this constraint is used, the lunch break defined in the Duration constraint must then terminate during the time window defined by Start and End.

Duration (*lunch.duration*): This constraint indicates the time allocated to the resource's lunch break. This duration is not considered by TourSolver as working time, but as break time. It is also important to note that it is deducted from the length of time set for the working day.

Example 1: possible instances found by TourSolver



Example 2:

A resource with:

Start time = 09:00;

Finish time = 18:00;

Lunch time = 01:00;

will be able to work (and fulfill visits) during a period of 8 hours (= 18 – 9 – 1).

Overtime

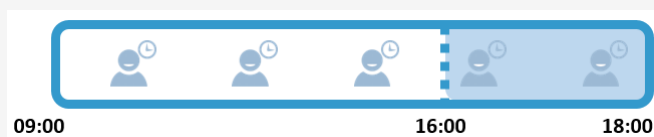
Number (*overtimeDuration[0]*)

This constraint indicates the authorized overtime hours. It indicates the duration of the first daily overtime time slot. When this value is indicated, the resource cannot work over and above the sum of the working time and the overtime time.

Format: *HH:MM*

Example:

In the case of a resource for whom the Work time = 09:00 and 16:00; If the Number = 02:00 then the length of the resource's working time is 9 hours (=16-9+2).



Overcost (*overtimePenalty[0]*)

This constraint allows you to indicate the excess cost of the hours worked during the first slot of daily overtime hours as recorded in the Number constraint.

Format: number

Example:

If the Hourly cost is 20 and the cost of the overtime hour is 25, then you can record Overcost = 5 (25-20).

For a resource for whom the constraints are the following:

Hourly cost = 20

Work time = 08:00 et 18:00

Number = 01:00

Overcost = 5

TourSolver can stipulate the resource works from 08:00am to 07:00pm, since one overtime hour is authorized, and calculates a Work cost of 225 (20*10*25*1).

Depending on the costs, TourSolver will calculate whether it is less costly to deploy another resource or to consume overtime hours for the same resource.

Hourly cost (*workPenalty*)

This constraint allows you to indicate the cost of an hour of work undertaken by the resource. The Hourly cost AND the Cost per km must not be null simultaneously.

Format: number

Example:

For a resource for whom the constraints are the following:

Hourly cost = 20

Work time = 08:00 et 18:00

Number = 01:00

Overcost = 5

The route TourSolver issues stipulates that the resource finishes at 04:00pm, so there are 8 working hours (4pm to 8pm), the total cost of the work amounts to 160 (20*8).

Cost per km (*travelPenalty*)

This constraint allows you to indicate the cost per journey unit (km, miles...) travelled by the resource.

Format: number

Example:

For example, the cost of 1km travelled costs 0.7, then the Cost per km = 0.7

Speed weighting (*speedAdjustment*)

If the travel times observed are generally speaking too low or too high, the user can correct them by specifying, for each resource, at what percentage of the speed defined in the road statistics they are really travelling. In this way the user can adapt the traffic data to the constraints for the resources while retaining the information about the slowing in speed as a function of the time and the direction of circulation.

Format: number (expressed as a percentage %)

As 100% is the default value and the reference value for the speed defined in the vehicle profile and/or given by the DataPack (road statistics).

Example 1:

The statistics indicate that the speed on the Avenue Aristide Briand at Bagneux is 50km/h. The user deems this time to be too quick, and assigns a value to the Speed weighting of 80, the resource drives 20% less quickly and so at 40 km/h instead of at 50 km/h.

Example 2:

The statistics indicate that the speed on Avenue Aristide Briand at Bagneux is 50 km/h. The user deems this speed to be too low, and enters a value of 120 for the Speed weighting constraint, so the resource drives 20% faster and so at 60 km/h instead of 50 km/h.

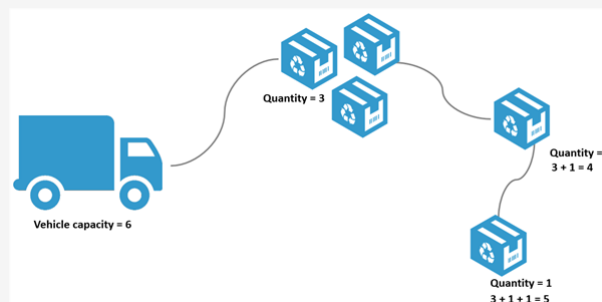
Vehicle capacity (*capacity[0]*)

This value represents the max capacity that a resource can deliver or collect. The user defines both units and quantity (kilos, m3, litres...) and ensures consistency with the visit Quality constraint.

Format : number

Example:

In the case of a parcel delivery, indicate the maximum number of parcels that the resource can carry (6 for 6 parcels), and in the case of a liquid cargo, indicate the number of litres (3000 for 3000 litres), in the case of bulky packaging indicate the volume (2 for 2m3).



Vehicle

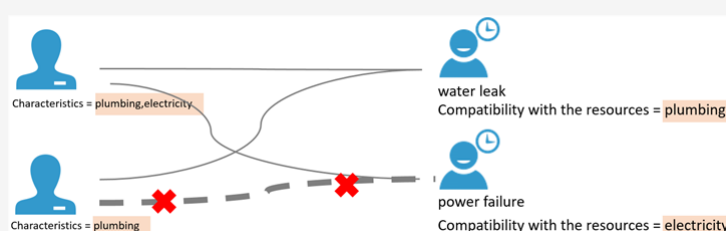
Characteristics (*providedSkills*)

This constraint designates the criteria required by the visit that the resource must follow to enable the visit to take place.

Format: character (expressed in list format, with the list items separated by commas.)

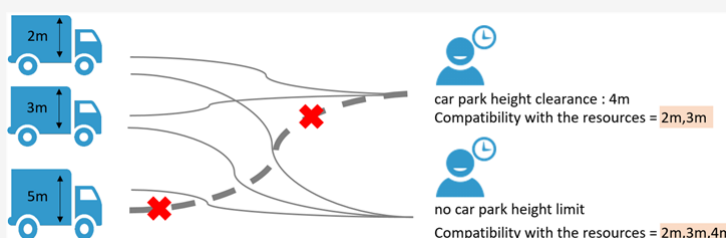
Example 1:

For a customer needing a maintenance intervention concerning the plumbing, you can indicate "plumbing". At least one resource must dispose of this characteristic to fulfill the customer visit.



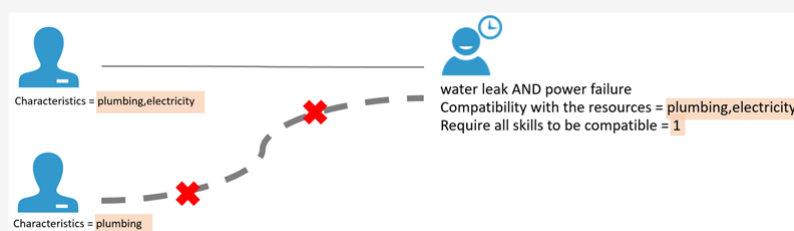
Example 2:

When access to a customer is limited to a height of 4m, we indicate "2m, 3m" since they can only receive delivery by a vehicle that is less than 4m in height. Only one of the resources fulfilling this criterion can visit this customer.

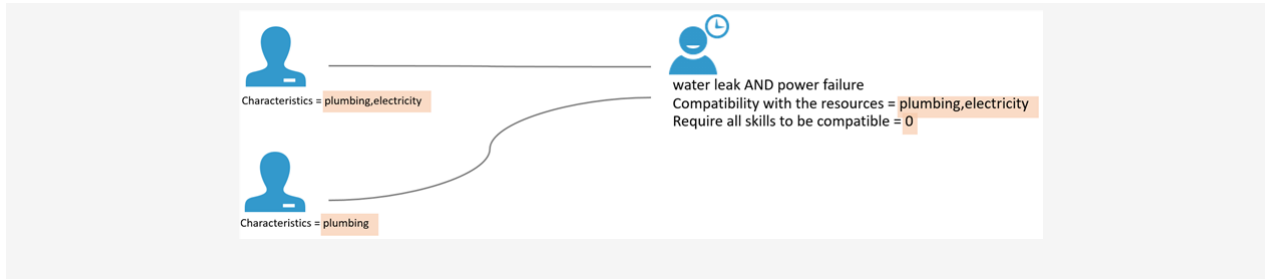


Example 3:

For a customer needing a maintenance intervention concerning the plumbing AND electricity, you can indicate "plumbing, electricity". At least one of the resources must fulfill both these criteria for the visit to take place. In this case, validate the constraint Require all skills to be compatible with a value of 1.



If the Require all skills to be compatible constraint = 0 then it will not be guaranteed that these 2 characteristics are taken into account.



Planning

Availability of the resource (*available*)

This constraint allows you to take the resource's availability into consideration without having to add or remove any data from the full data set.

If the button is set to Available the resource can execute the route.

If the button is set to Unavailable, the resource cannot undertake the route.

The default value is Available.

Work time 2 to 4 (*otherWorkStartTime[0]...[2],otherWorkEndTime[0]...[2]*)

These constraints define other time windows that will apply to working days defined under Worked days 2 to 4.

Format: HH:MM

Example 1:

The resource can work from Monday to Thursday between 09:00am and 05:00pm and on Friday between 09:00am and 12:00pm. For this we use the constraint:

Work time 1 = 09:00 and 17:00;

Worked days 1 = 1-4;

Work time 2 = 09:00 and 12:00;

Worked days 2 = 5.

Example 2:

The resource can work on Mondays between 07:00am and 05:00pm and on Wednesdays between 09:00am and 06:00pm, in this case:

Work time 1 = 07:00 and 17:00;

Worked days 1 = 1;

Work time 2 = 09:00 and 18:00;

Worked days 2 = 3.

- ❗ For a given day, only a time window can be defined. It is therefore not possible to set this example: "the resource can work in the morning between 09:00am and 12:00pm and in the afternoon between 02:00pm and 05:00pm without any lunch break".

Worked days 1 to 4 (*workingDays,otherWorkDays[0]...[2]*)

This constraint contains all Worked days (up to 64 days) to which the time window defined by the Work time constraints applies.

Format: character

These days can be defined one by one (1,4), as integers (1,2, ...64) or in date format (14/05/2016 => 17/07/2016).

Example 1:

Worked days = 1,2,5 (or 14/05/2016,15/05/2016,18/05/2016) means that the resource can perform interventions or visits on days 1, 2 and 5 in the planning (or on 14/05/2016, 15/05/2016 and 18/05/2016) at the times defined by the Work time constraints. Worked days = 1-5 (or 14/05/2016 => 18/05/2016) means that the resource can perform visits on all days 1, 2, 3, 4 and 5 of the planning (or between the 14/05/2016 and the 18/05/2016) at the times defined by the Work time constraints.

Example 2:

The resource can work on Mondays between 07:00am and 05:00pm and on Wednesdays between 09:00am and 06:00pm: Work time = 07:00 and 17:00, Worked days = 1 and Work time 2 = 09:00 and 18:00, Worked days 2 = 3

Example 3:

The resource can work from Monday to Friday between 07:00am and 05:00pm and on Saturday between 09:00am and 12:00pm: Work time = 07:00 and 17:00, Worked days = 1-5 and Work time 2 = 09:00 and 12:00, Worked days 2 = 6

- ⓘ Caution: this constraint must be set up in correspondence with the visit's Opening days constraints and must use the same format (date or number). None of the Opening days dates can be earlier than the latest date of the Worked days.

Automatic start time (*optimumStartTime*)

This constraint can adjust the resource's departure time by taking into account the visit's opening hours.

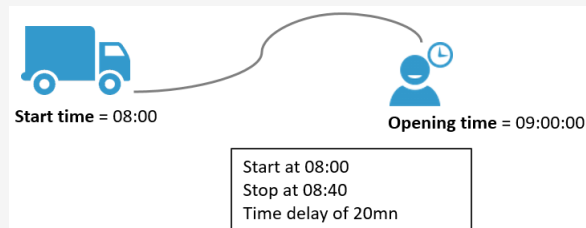
If the button is set to Disabled, the resource starts their route at the time indicated by the Work time constraint whatever the opening time set for the first visit (waiting time is possible);

If the button is set to Enabled, the resource adapts their departure time to optimize the route and reduce the possible waiting time, if there is one, as a function of the opening time set for the first visit.

In any case, the resource cannot leave before the time indicated by their Work time constraint. The end time is shifted by the same amount of time, except where the Daily work time constraint is also used.

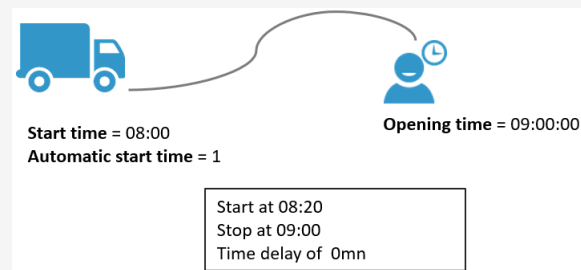
Example 1:

If Automatic start time is disabled



Example 2:

If Automatic start time is enabled



Daily work time (*dailyWorkTime*)

This constraint indicates the maximum duration of time that the resource can spend each day, excluding any overtime hours defined if these are being used.

Format: HH:MM

Weekly working time (*weeklyWorkTime*)

This constraint indicates the maximum time as working hours that the resource can perform each week. This weekly quota must then be recorded within the resource's working time windows: adding overtime hours to this duration is not possible.

Format: HH:MM

Note that the notion of week used is: in the case of an utilisation of a date format in the Worked days constraint, that of a calendar week / or if not, that defined by the planner (7-day interval).

Overtime 2

Number (*overtimeDuration[1]*)

This constraint indicates the authorized overtime duration. It records the duration of the second daily overtime time window. When this value is indicated, the resource cannot work over and above the sum of the work time and overtime duration.

Format: HH:MM

Example:

In the case of a resource for whom the Worked days = 09:00 et 16:00 and Number overtime 1 = 02:00, if Number overtime 2 = 01:00 then the overall length of time the resource will be working is 10h (=16-9+2+1).



Overcost (*overtimePenalty[1]*)

This constraint allows you to indicate the overtime cost of hours worked during the second slot of daily overtime hours as recorded under the Number overtime 2 constraint.

Format: number

Example: if the Hourly cost is 20 and the cost of an overtime hour is 30, then the Overcost 2 = 10 (30-20).

For a resource for whom the constraints are the following:

Hourly cost = 20

Work time = 08:00 et 18:00

Number = 01:00

Overcost = 5

TourSolver can deploy the resource from 08:00am to 08:00pm because two overtime hours have been authorized, and it will calculate a work cost of 255 (20*10+25*1+30*1).

Depending on the costs, TourSolver will calculate whether it is less costly to deploy another resource or to consume overtime hours for the same resource.

Briefing duration (*briefingDuration*)

This constraint indicates a fixed duration to add before the resource's departure to fulfill the route. This time is applied to the departure location of the resource.

Format: HH:MM:SS

Example:

Before departing on their routes, a driver must wash the vehicle, a technician must get changed in the staff changing room, a delivery person must retrieve the delivery notes related to their route.

Debriefing duration (*debriefingDuration*)

This constraint indicates a fixed duration to add following the return of the resource from the day's routes. This time is applied to the arrival location of the resource.

Format: HH:MM:SS

Example:

Returning from their route, the driver must wash the vehicle, a technician must change in the staff changing room, and a driver must drop off the signed delivery notes.

Tour

Start at first visit (*openStart*)

This constraint allows you to indicate whether the resource starts their route at the first visit or at the address indicated as their geolocalised address.

If the value is set to Disabled, the cost of the route is calculated from the localisation of the resource.
 If the value is set to Enabled, the cost of the route is calculated from the first visit.

The default value is Disabled.

If the value of the First stop at depot constraint is Enabled, the cost calculated starts at the site where loading is to take place.

End at last visit (*openStop*)

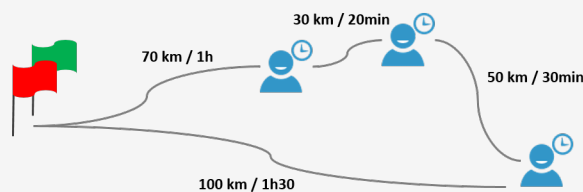
This constraint allows you to indicate whether the resource finishes their route at the last visit or at the address indicated for the resource's geolocalisation.

If the button is set to Disabled, the cost of the route is calculated up until the resource's return to geolocalisation address.

If the button is set to Enabled, the cost of the route is calculated up until the resource's return to the last visit.

The default value is Disabled.

If the value of the Last stop at depot constraint is Enabled, the cost calculated finished at the site where reloading is to take place.



Example 1: Start at first visit = Disabled and End at last visit = Disabled

The journey time of this route is 3h20 (1h+20m+30m+1h30) and distance is 250kms (70+30+50+100).

Example 2: Start at first visit = Enabled and End at last visit = Enabled

The journey time of this route is 50 minutes (20m+30m) and distance is 80kms (30+50).

It is possible to dissociate the start and end constraints.

Measure distance from first visit

This constraint allows you to count or not in the optimisation the distance between the resource start location and the first visit (resource location or start point if is defined).

Format: binary

If the value is set to Disabled, the cost km of the route is calculated from the localisation of the resource. If the value is set to Enabled, the cost km of the route is calculated from the first visit. The default value is Disabled.

- ! If the value of the First stop at depot constraint is Enabled, the cost km calculated starts at the site where loading is to take place.
If the value of the Briefing duration is used, the cost km calculated starts at the stop following the briefing.
If the value of the Start at first visit constraint is Enabled, this constraint is disabled.

Stop distance measurement at last visit

This constraint allows you to count or not in the optimisation the distance between the last visit and resource stop (resource location or stop point if is defined).

Format: binary

If the value is set to Disabled, the cost km of the route is calculated up until the resource's return to geolocalisation address. If the value is set to Enabled, the cost km of the route is calculated up until the resource's return to the last visit. The default value is Disabled.

- ! If the value of the Last stop at depot constraint is Enabled, the cost km calculated stops at the site where reloading is to take place.
If the value of the Debriefing duration is used, the cost km calculated stops at the stop preceding the debriefing.
If the value of the End at last visit constraint is Enabled, this constraint is disabled.

Measure time from first visit

This constraint allows you to count or not in the optimisation the time between the resource start location and the first visit (resource location or start point if is defined).

Format: binary

If the value is set to Disabled, the hourly cost of the route is calculated from the localisation of the resource.

If the value is set to Enabled, the hourly cost of the route is calculated from the first visit.
The default value is Disabled.

- ❗ If the value of the First stop at depot constraint is Enabled, the hourly cost calculated starts at the site where loading is to take place.
If the value of the Briefing duration is used, the hourly cost calculated starts at the stop following the briefing.
If the value of the Start at first visit constraint is Enabled, this constraint is disabled.

Stop time measurement at last visit

This constraint allows you to count or not in the optimisation the time between the last visit and resource stop (resource location or stop point if is defined).

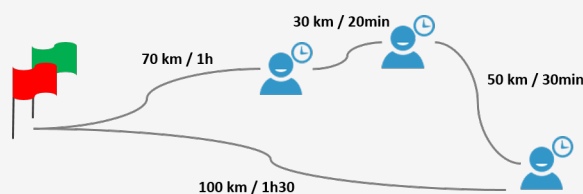
Format: binary

If the value is set to Disabled, the hourly cost of the route is calculated up until the resource's return to geolocalisation address.

If the value is set to Enabled, the hourly cost of the route is calculated up until the resource's return to the last visit.

The default value is Disabled.

- ❗ If the value of the Last stop at depot constraint is Enabled, the hourly cost calculated stops at the site where reloading is to take place.
If the value of the Debriefing duration is used, the hourly cost calculated stops at the stop preceding the debriefing.
If the value of the End at last visit constraint is Enabled, this constraint is disabled.



Example 1: Open time start and stop = Disabled and Open distance start and stop = Enabled

The journey time of this route is 3h20 (1h+20m+30m+1h30) and distance is 80kms (30+50).

Example 2: Open time start and stop = Enabled and Open distance start and stop = Disabled

The journey time of this route is 50 minutes (20m+30m) and distance is 250kms (70+30+50+100).

Maximum daily distance (*maximumDistance*)

This constraint allows you to define the maximum distance a resource may travel in one working day.

Format: number

Maximum visits per day (*maximumVisits*)

This constraint designates the maximum number of visits a resource can perform during one working day.

Format: number

Loading/Unloading

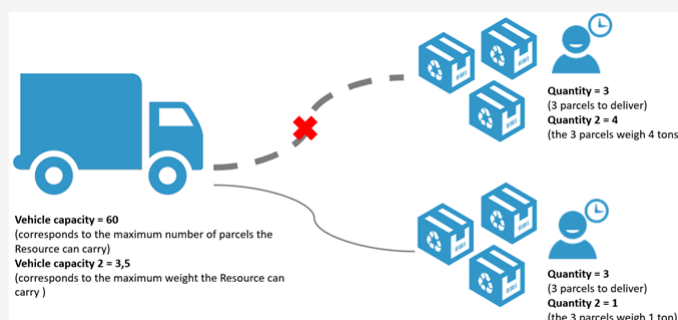
Vehicle capacity 2 to 24 (*capacity[1],...capacity[23]*)

In addition to the Vehicle capacity constraint, TourSolver suggests 24 other measurement constraints to deliver or collect. Each of them should be put into correspondence with the Visit's constraints Quantity, Quantity 2, ..., Quantity 24 for different delivery units to be taken into account.

Format : number

Example:

Deliveries or collections of heavy and voluminous objects. Use Quantity and Vehicle capacity for the weight, Quantity 2 and Vehicle capacity 2 for the volume. The loading on the part of resources and the delivery to, or collection from customers will be optimised as a function of the double constraint of weight and volume.



Global capacity (*globalCapacity*)

This value represents the total capacity a resource can deliver or collect when several quantities are used for visits and sites.

Format: number

Example:

For a delivery of a cargo of vegetables, for example, we need to indicate for the resource:

- Vehicle capacity 1 = 150 (kg of carrots)
- Vehicle capacity 2 = 50 (kg of pumpkins)

- Vehicle capacity 3 = 100 (kg of potatoes)

That is, a total of 300kg of vegetables overall.

and the Global capacity = 200

In this instance, the resource can only carry a maximum of 200kg of vegetables overall, of all types.

Use all capacities (*useAllCapacities*)

If the button is set to Disabled, only the Global capacity is taken into account.

If the button is set to Enabled, all capacities, including the Global capacity, are taken into account.

Format: binary

Example:

For a delivery of a cargo of vegetables, for example, we need to indicate for the resource:

- Vehicle capacity 1 = 150 (kg of carrots)
- Vehicle capacity 2 = 50 (kg of pumpkins)
- Vehicle capacity 3 = 100 (kg of potatoes)
- Global capacity = 200 (kg)

If Use all capacities is set to Disabled, the value 200 for the Global capacity alone must not be exceeded, and the other capacities are not taken into account. The resource can carry 30kg of carrots, 60kg of pumpkins and 110kg of potatoes.

If Use all capacities is set to Enabled, all capacities must be respected. The resource can carry 50kg of carrots, 50kg of pumpkins and 100kg of potatoes.

Minimum quantity to deliver (*minimumQuantity*)

This constraint indicates the minimum quantity that the resource must deliver. If this quantity is not null, then the resource can perform visits only if their quantity to deliver or pick-up is greater than the value recorded under this constraint.

Format : number

Example:

Taking an example of a resource for whom the Minimum quantity to deliver is 2, and a customer A for whom the Quantity is 4, a customer B for whom the Quantity is 1, and a customer C for whom the Quantity is 2, only customer A will be visitable since $4 > 2$.

Loading/Unloading duration at site (*fixedLoadingDuration*)

This is the minimum time, specific to the resource, that they must spend at the site for loading or unloading, whatever the site. To this time can be added the times entered under the following constraints:

Loading/Unloading duration per unit at site (resources), Fixed loading/unloading duration (sites) and Loading/Unloading duration per unit (sites).

Format: HH:MM:SS

Example:

For a collection, the resource will stay for the time indicated under Loading/Unloading duration at site, independently of the quantity to be collected. This time may correspond, for example, to the time it needs to operate the vehicle's tailgate, that needs to be added to the reference time to spend at the customer's premises as recorded under the Visit duration constraint.

Loading/Unloading duration per unit at site (*loadingDurationPerUnit*)

This value corresponds to the time needed specifically by the resource to load/unload one unit of product at the site. This value is multiplied by the quantity of product to load or unload, depending on the type of delivery or pick-up operation. This constraint applies only to the Quantity 1 constraint for visits, and not to the 23 other possible Quantities.

Format: HH:MM:SS

Example:

In the case of a collection of 10 bottles of gas: the estimated loading time for a single bottle of gas is 3 minutes. A value of 00:03:00 should be entered for the Loading/Unloading duration per unit at site constraint. The resource will therefore take 30m to complete the collection (3m x 10 bottles).

Loading/Unloading duration for each visit (*fixedUnloadingDuration*)

This is the minimum time specific to the resource, that they must spend for a loading/unloading operation whoever the customer is, and so, a minimum visit time at the customer's premises. To this time can be added the times entered under the following constraints: Loading/Unloading duration per unit for each visit (resources), Visit duration (visits) and Loading/Unloading duration per unit (visits).

Format: HH:MM:SS

Example:

For a delivery, the resource will remain at the site for the time indicated under Loading/Unloading duration for each visit, independently of the quantity to collect. This time may correspond for example to the time needed to operate the vehicle's tailgate... this is added to the set time to call at the customer's as recorded in the Visit duration constraint.

Loading/Unloading duration per unit for each visit (*unloadingDurationPerUnit*)

This value corresponds to the time needed, specific to the resource, to deliver or collect one unit of product during a visit. This value is multiplied by the Quantity 1 value of the visit. This constraint is applied only to the Quantity 1 constraint of the visit, and not to the 23 other possible Quantities.

Format: HH:MM:SS

Example:

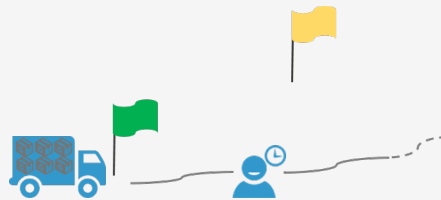
In the case of a delivery of 10 parcels to a customer: the estimated unloading time for a single parcel is 3 minutes. A value of 00:03:00 (or 180 seconds) should be entered for the Loading/unloading duration per unit for each visit and a value of 10 in the Quantity constraint: the resource will therefore spend 30 minutes for the delivery (3 min x 10 parcels).

First stop at depot (*loadBeforeDeparture*)

This constraint allows you to indicate whether the resource partially loads or whether they must stop to load at the site address, or if not, at the address where they are located before the first visit.

Example:

If the button is set to Disabled, the resource partially loads to perform the first visit.



If the button is set to Enabled, the resource must load before fulfilling the first visit.

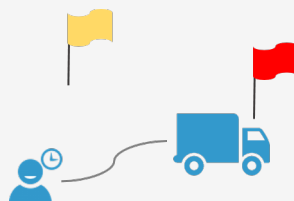


Last stop at depot (*loadOnReturn*)

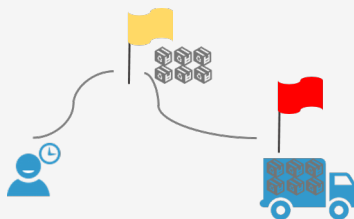
This constraint allows you to indicate whether the resource terminates unloaded (empty) or whether it must stop to reload at the site address or if not, at the address where they are located after the last visit.

Example:

If the button is set to Disabled, the resource does not reload after their last visit.

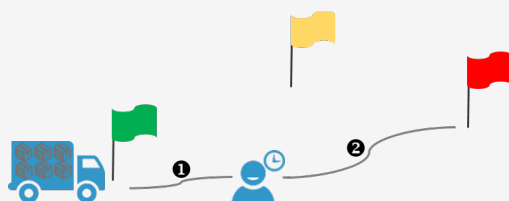


If the button is set to Enabled, the resource must reload after their last visit.

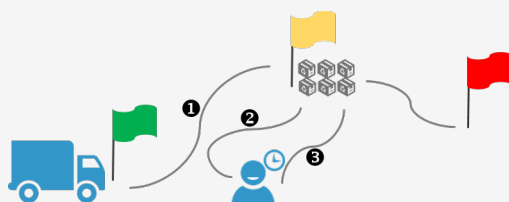


Example of utilisation of the First stop at depot and Reload after return constraints.

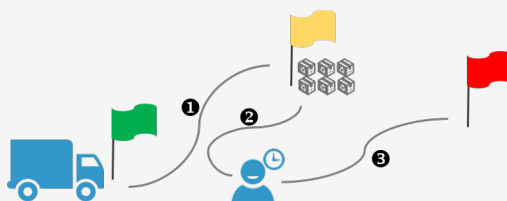
If First stop at depot and Last stop at depot are disabled



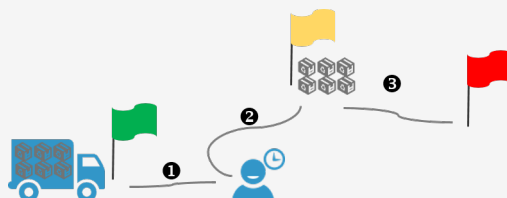
If First stop at depot and Last stop at depot are enabled



If First stop at depot is enabled and Last stop at depot is disabled



If First stop at depot is disabled and Last stop at depot is enabled



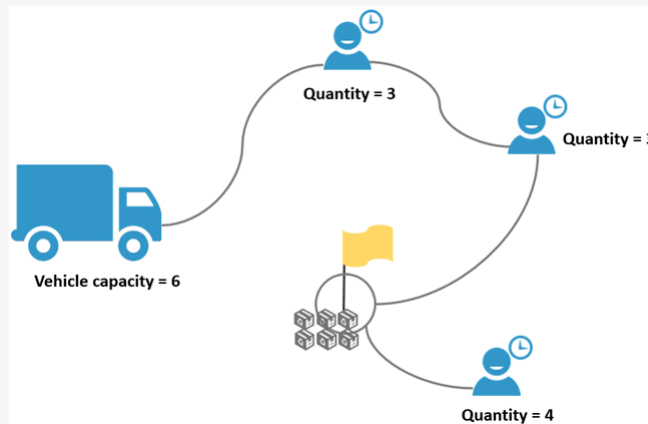
Reload/Unload (*noReload*)

This constraint allows you to forbid a resource to reload at the site during a route. If the First stop at depot and Last stop at depot constraints are used, the resource will never reload at the site during a route. In this case, the resource will load once at the start or at the end, or twice, at the start and at the end.

If the button is set to Forbidden to forbid the reloading.
If the button is set to Allowed to authorize the reloading.
The default value is Allowed.

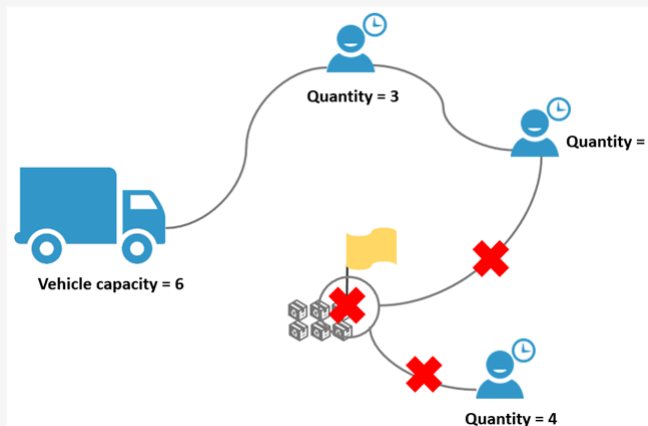
Example:

If Reload/unload is allowed



All customers can be visited even if the total quantity to deliver exceeds the initial capacity of the resource, it reloads at the depot visited during the route.

If Reload/unload is forbidden



In the case where a customer cannot be visited by this same resource since the maximum capacity of the resource has been reached and they cannot reload at the depot.

Maximum number of reloads/unloads (*maximumReloads*)

This constraint allows you to limit the number of reloads when this is authorized in the Reload/Unload constraint. It is a maximum number of reloadings that a resource can perform per day.

Format: number

Exceeded reloads/unloads penalty (*maximumReloadsPenalty*)

This constraint allows you to give an additional cost applied per reload over and above the value expressed under the Maximum number of reloads/unloads constraint.

Format: number

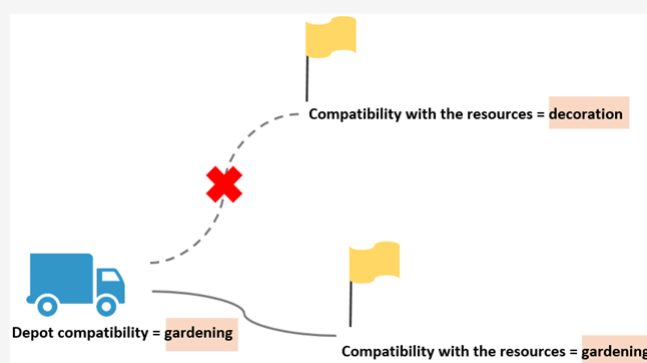
Depot compatibility (*providedProducts*)

This constraint designates the criteria required by the resource for them to be able to arrive at a site.

Format: character, expressed in the form of a list of words separated by commas.

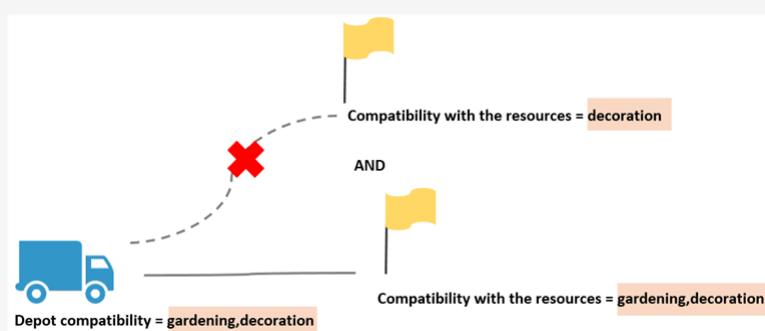
Example 1:

For a resource needing to call in at a depot to withdraw an order concerning gardening tools, you can indicate "gardening". At least one depot must dispose of this criterion for the resource to be able to call in.

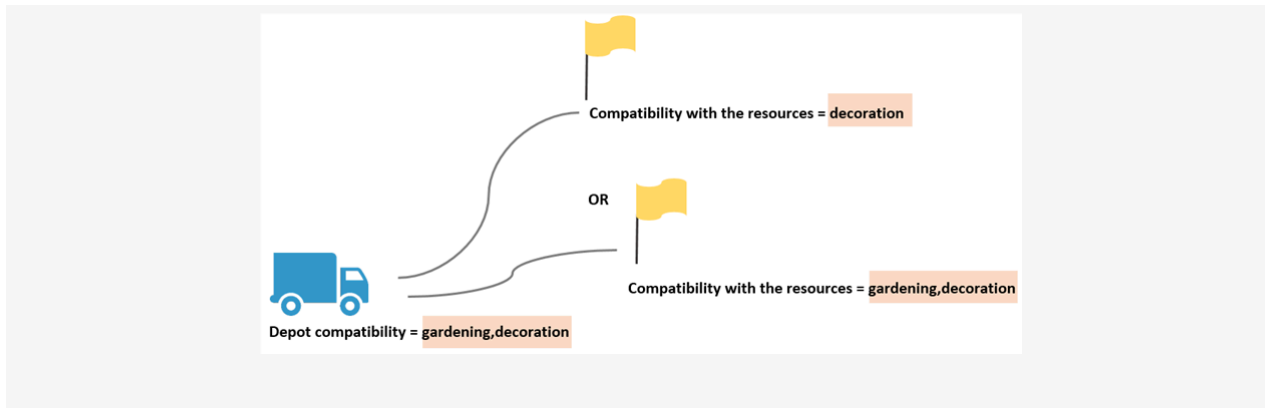


Example 2:

For a resource needing to call in at a depot to collect an order concerning gardening tools AND another order concerning interior decoration, you can indicate "gardening, decoration". At least one depot must dispose of these 2 compatibilities for the resource to be able to call in there. You should therefore put a value of 1 in the All compatibilities constraint.



If the All compatibilities constraint = 0, then the taking into account of the 2 compatibilities will not be requested.



Costs

This part concerns the cost constraints. The TourSolver engine bases its calculations on these constraints to optimize the routes. Its first priority is to suggest the route with the lowest total cost, while respecting the other constraints. This is why it is important to consider these constraints, when deliberating about cost, we are talking above all about weight. For the TourSolver application, there is no notion of monetary cost, but of a weighting.

Pay whole day (*payWholeDay*)

This constraint allows you to indicate whether all the times in the day are accounted for or just those used.

If the button is set to Disabled, only the worked hours are used for the calculation.

If the button is set to Enabled, all the hours in the day are used for the calculation.

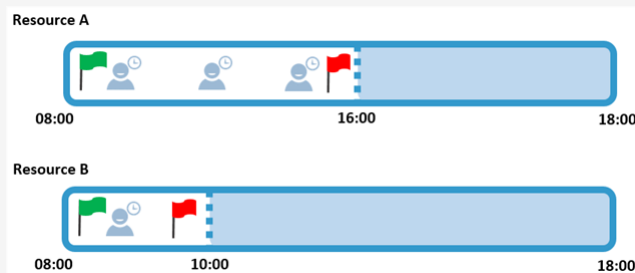
This constraint has an impact on the way the resources' plannings are filled.

Example:

For 2 resources A and B: Hourly cost = 20 Work time = 08:00 and 18:00

If Pay whole day is disabled

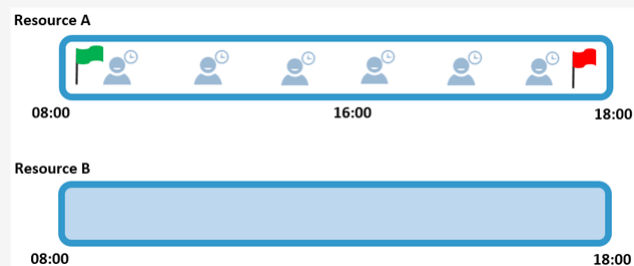
TourSolver may suggest this planning:



In this case, resource A will therefore work 8 hours and resource B will work 2 hours. This means a total cost of 200 is calculated by TourSolver ($8 \cdot 20 + 2 \cdot 20$).

If Pay whole day = 1

TourSolver may suggest this planning:



Resource A works therefore 10 hours and resource B works 0 hours. This means the total cost is 200 (10 * 20) as calculated by TourSolver and only one single resource is used.

Fixed cost of the resource used (*useInPlanningPenalty*)

This constraint allows you to indicate a fixed daily employment cost for the resource, that will be applied to each day the resource is deployed on a route.

Format: number

Example:

The delivery company fleet is made up of an existing resource A and a commissioned (subcontracted) resource B. In this case, the company will prefer to use their own resource with the defined Hourly cost and Cost per km constraint values, and then further down the line the commissioned resource who will have a defined daily utilisation rate or Fixed cost of the resource used.

For resource A:

Hourly cost = 20

Journey cost = 1

Fixed cost of the resource used = 0

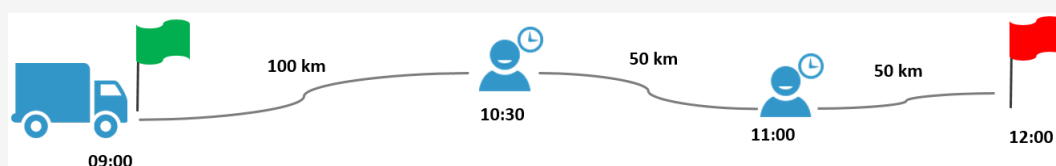
For resource B:

Hourly cost = 0

Journey cost = 0

Fixed cost of the resource used = 600

Route 1/2



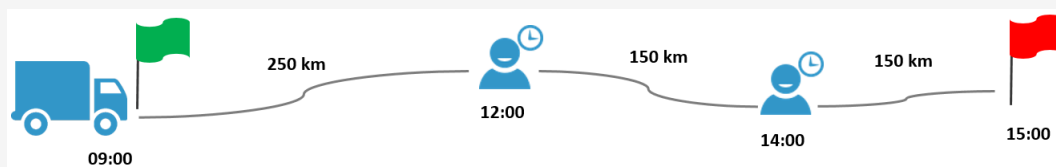
This route costs 3 hours of work (12-9) and 200 kms (100+50+50). To deliver to these 2 customers, TourSolver will calculate which of the resources A and B will be the least costly.

Resource A = 240 (20*2h + 1*200km + 0)

Resource B = 600 (0*2h + 0*200km + 600)

In view of the total route cost, TourSolver will choose to use Resource A to fulfill this route.

Route 2/2



This route costs 6 hours of work (15-9) and 550 kms (250+150+150). To deliver to these 2 customers, TourSolver will calculate which of the two resources A and B will be least costly to use:

Resource A = 670 (20*6h + 1*550km + 0)

Resource B = 600 (0*6h + 0*550km + 600)

In view of the total route cost, TourSolver will choose to use Resource B to fulfill this route.

Resource daily cost not used (*nonUsePenalty*)

This constraint allows you to indicate a fixed daily non-use cost for the resource, that will be applied each day where the resource is not used on a route. This constraint can be used to integrate within the optimisation calculation the costs incurred by a company even when the resource is immobilised (depreciation, parking, ...).

Format: number

Example:

For Resource A:

Resource daily cost not used = 0

And for Resource B:

Resource daily cost not used = 600

1) If TourSolver chooses to deploy Resource A, then this route will cost 600 (0+600) since the non-utilised Resource B, will nonetheless still cost 600.

2) If TourSolver chooses to deploy Resource B, then this route will cost 0 (0+0) since the non-utilised Resource A will not cost anything, unlike Resource B.

The result will be that TourSolver will select the second solution.

Resource request cost (*usePenalty*)

This constraint allows you to indicate a fixed cost for deploying the resource for the whole of the route, which will be applied as soon as the deployment takes place. Unlike the Fixed cost of the resource used

this constraint does not apply to each day worked by the resource, but applies as from the first day worked.

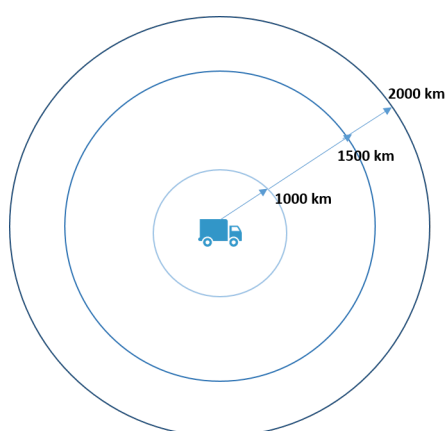
Format: number

Cost km 2 to 4

Format: number

Threshold [distance_2...4]: This constraint allows you to indicate a distance threshold, above which, a different cost will apply.

Cost [penalty_2...4]: This constraint allows you to indicate the cost per journey unit (kms, miles...) travelled by the resource over and above a certain distance defined under the Threshold constraint.



Example:

For a company the fleet of which is made up of commissioned resources travelling numerous kilometers.

Cost per km = 1

Cost km 2 Threshold = 1000

Cost per km 2 = 1,5

Cost km 3 Threshold = 1500

Cost per km 3 = 2

If this transporter travels 2000 km during the week, then the total kilometer cost = 4000 (2000 x 2). This is the Cost per km 3 = 2, as indicated for Cost km 3 Threshold that is applicable because 2000 > 1500.

Fixed visit cost (*penaltyPerVisit*)

This constraint allows you to define, for each resource, a fixed cost to add to each visit fulfilled by the resource. In this way, TourSolver will add this cost to the global cost for the resource.

Format: number

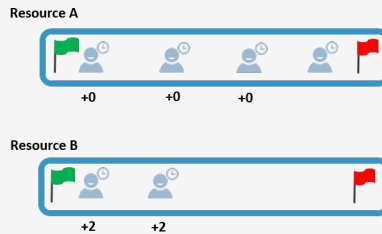
Example:

For Resource A, Fixed visit cost = 0,

For Resource B, Fixed visit cost = 2.

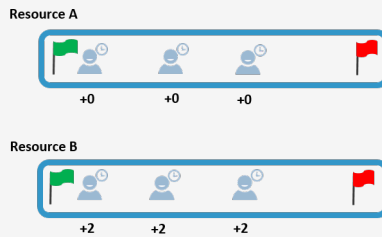
TourSolver will seek to fill the planning of Resource A first, since the cost of the route is lower than the equivalent route of Resource B, who has a global cost that is incremented as a function of the number of visits.

Example of a route calculated and selected by TourSolver:

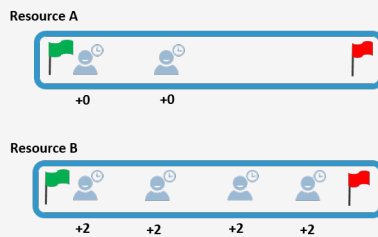


The overall cost of this route is calculated at 4 (2+2+0+0+0+0).

Example of routes not selected by TourSolver:



The global cost of this route is 6 (2+2+2+0+0+0).



The global cost of this route is 8 (2+2+2+2+0+0).

Nights

Max consecutive nights out (*maxNightsOutPerJourney*)

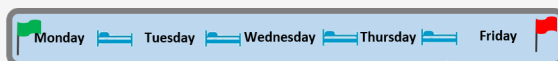
This constraint allows you to indicate the maximum number of consecutive nights away without returning to base.

Format: number

Example:

A salesperson works travelling routes away from Monday until Friday without returning to base.

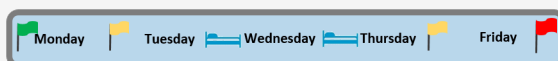
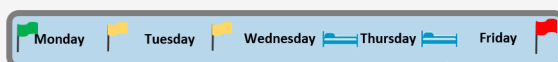
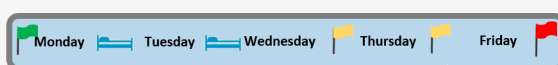
Worked days = 1-5 et Max consecutive nights out = 4



A salesperson working away on routes from Monday to Friday but limited to 2 nights away without returning to base.

Worked days = 1-5 et Max consecutive nights out = 2

TourSolver will suggest the following routes:



Night out cost (*nightPenalty*)

This constraint allows you to specify a global cost for a night away. TourSolver will take this cost into account to determine whether it is preferable to have the resource sleep the night in paid accommodation or to have them return to base. The lower the cost, the more attractive in cost terms the night away will be.

Format: number

Max drive time back before night out (*overnightMinDriving*)

This constraint allows you to define the maximum drive time enabling the resource to return to base rather than take the night away.

Format: HH:MM

Example 1:

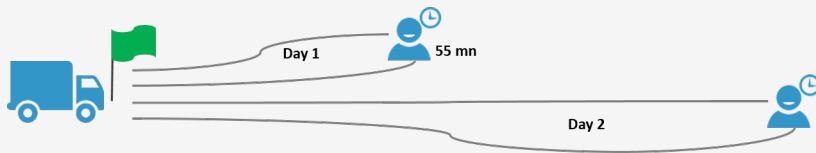
Max drive time back before night out = 00:30 indicates that if the resource is more than 30 minutes away from base, the night will be spent in external accommodation. Conversely, if the drive time back to base is less than 30 minutes, they will return to base.



In this case, the resource sleeps the night in external accommodation since it is more than 30 minutes' drive from the base.

Example 2:

Max drive time back before night out = 02:00 indicates that if the resource is more than 2 hours from base, they will sleep in external accommodation. Conversely, if the drive time back to base is less than 2 hours, the resource will return to base.



In this case, the resource will not stay the night away, since they are less than 2 hours drive away from base.

Driving legislation

Max drive duration without break [`legalMaxDriveDuration`]

This constraint allows you to indicate the maximum drive time that the resource can undertake without a break.

Format: HH:MM

Example:

A resource may not drive more than 4 hours and 30 minutes continuously without a break, and so the Max drive duration without break = 04:30

Drive break duration (`legalDriveRestDuration`)

This constraint allows you to indicate the duration of the break or pause a resource should take when the resource has reached the maximum number of continuous driving hours before resuming their journey.

Format: HH:MM

Example:

A resource may not drive again if they have not taken a break lasting 45 minutes, and so Drive break duration = 00:45.

Drive break min duration (`legalMinRestDuration`)

This constraint allows you to indicate the Drive break min duration for it to be taken into account as rest time during a driving session.

Format: HH:MM

The Drive break duration constraint is not considered by TourSolver as working time. The duration of a working day is prolonged by the equivalent length of time.

Example:

A break is considered as a rest when it lasts upwards of 15 minutes, and so Drive break min duration = 00:15

Example: illustration of the Legal max driving duration + Legal driving break duration + Legal min rest duration constraints.

If Max drive duration without break = 04:30;

If Drive break duration = 00:45;

If Drive break min duration = 00:15;

This means the resource can drive for 4 hours and 30 minutes if they take 45 minutes break. This break can take the form of 3 15-minute pauses or one pause of 30 minutes and then another of 15 minutes, for example. Conversely, if the resource has taken 2 breaks of 20 minutes, and one of 10 minutes, they will be obliged to retake a break at least 15 minutes long in order to be allowed to drive 4 hours and 30 minutes, since the 10-minute break is not counted as a pause, since the duration is less than 15 minutes.

Drive break at customer (*driveRestAtCustomer*)

This constraint allows you to indicate whether the time spent at a visit is counted as rest time.

If the button is set to Disabled, the time is not counted as rest time;

If the button is set to Enabled, the time is counted as rest time.

Drive break at depot (*driveRestAtDepot*)

This constraint allows you to indicate whether the time spent at the site for a reloading operation is considered as rest time.

If the button is set to Disabled, the time is not counted as rest time;

If the button is set to Enabled, the time is counted as rest time.

Daily drive time (*legalDailyDriveDuration*)

This constraint allows you to indicate the maximum drive time that the resource can perform without staying away overnight for one day. This constraint allows you to indicate the maximum accumulated drive time the resource can undertake without staying the night away in one day. At the end of this driving time as specified, the resource must take a break of duration as specified under Legal night rest duration.

Format: HH:MM

Example:

A resource cannot drive more than 9 hours without a night break, so Daily drive time = 09:00

Night out rest duration (*legalDailyRestDuration*)

This constraint allows you to indicate the duration of the night break taken by a resource, when the Daily drive time is exceeded.

Format: HH:MM

Example:

A resource cannot drive again if they have not had a night of at least 4 hours, so Night out rest duration = 04:00.

Configuring sites

- ❗ In this documentation, keywords in the default terminology are used as defined in the *My activity* page of the interface.
So, although we will use the word "site" here, the names of the constraints may well vary as a function of the terminology chosen when configuring the application.

- ❗ After each constraint name, the brackets contain the equivalent name assigned to this constraint under [TourSolver Cloud API](https://geoservices.geoconcept.com/ToursolverCloud/api-book.html) [https://geoservices.geoconcept.com/ToursolverCloud/api-book.html].

Site name (*id*)

This value allows you to identify the site. The name may be a patronym, the name of the town, or an identifier...

Format: character

Example:

Depot A, BAGNEUX, Agence Nantes3

Assigning a value to this constraint is mandatory.

It is recommended that this value is unique.

Opening times 1 (*timeWindow[0].beginTime,timeWindow[0].endTime*)

These constraints allow you to define a time window in which resources can go back to the site. Opening times 1 corresponds to the time from which the resource can start to (re)load and Closing times 1 the time beyond which the resource cannot (re)load. This time window applies to all days UNLESS Opening days is defined. In this case, the time window is applied only to this value.

Format: HH:MM

A (re)load operation cannot start before the start time. A (re)load operation started in the time window can continue beyond the end time. If you want the (re)load operation to finish before a certain time you will need to reduce this time by the (re)loading time (if this time is fixed) and define this new time under the closing times 1 constraint.

If a time window is not documented for a site, the resource can perform a (re)load operation at any time during any Opening days defined and within the limit set for the resource's working hours. You can define up to 4 different time slots (and their associated days) for each site.

- ❗ Time windows defined as overlapping over several days are not handled. The user cannot use a time window starting at 11:00pm and finishing at 03:00am the next day, but a bypass solution or fix does exist.

Example 1:

Opening times 1 = 07:30 and Closing times 1 = 11:30 define a time window of 4 hours during which the resource can (re)load.

Example 2:

The resource can undertake the (re)load at the depot between 11:00pm and 03:00am In this case, you will need to shift all the times (in one direction or the other) in order for them to be all contained in one day and to restore the sequencing of the route. Opening times 1 = 19:00 and Closing times 1 = 23:00 define a time window of 4 hours during which the resource can undertake the (re)load operation.

Following the optimisation, you need to shift all the visit and (re)load times again to restore the original time window. This (re)load operation was planned by TourSolver at 09:00pm, and in reality it will be undertaken at 01:00am (=23:00+2h). You will therefore need to add the constant of 2h to all the calculated times given.

Opening times 2 (*timeWindow[1].beginTime,timeWindow[1].endTime*)

These constraints allow you to define a second time window which will apply to visit days defined under Opening days 2. Opening times 2 corresponds to the time from which the resource can undertake the (re)loading operation and Closing times 2 the time beyond which the resource can no longer undertake the (re)load operation.

Format: HH:MM

Example 1:

The resource can undertake the (re)load operation at the depot between 09:00am and 12:00pm and in the afternoon between 02:00pm and 05:00pm. For this, we use the constraint Opening times 1 = 09:00, Closing times 1 = 12:00 and Opening times 2 = 14:00, Closing times 2 = 17:00

Example 2:

The depot is open on Mondays between 07:00am and 05:00pm and on Wednesdays between 09:00am and 06:00pm, in this case, Opening times 1 = 07:00, Closing times 1 = 17:00, Open days 1 = 1 and Opening times 2 = 09:00, Closing times 2 = 18:00, Opening days 2 = 3

Opening times 3 (*timeWindow[2].beginTime,timeWindow[2].endTime*)

These constraints allow you to define a second time window that applies to visit days defined under Opening days 3. Opening times 3 corresponds to the time from which the resource can perform the

(re)load operation and Closing times 3 the time beyond which the resource can no longer perform the (re)load operation.

Opening times 4 (*timeWindow[3].beginTime,timeWindow[3].endTime*)

These constraints allow you to define a second time window that applies to visit days defined under Opening days 4. Opening times 4 corresponds to the time from which the resource can perform the (re)loading operation and Closing times 4 the time beyond which the resource can no longer perform the (re)loading operation.

Opening days 1 (*openingDays[0]*)


This constraint contains all the possible visit days for calling in at the site (up to 64) to which the time window defined by the Opening times 1 and Closing times 1 constraints applies.

Format: character

These days can be defined one by one (1,4), as integers (1,2, ...64) or in date format (14/05/2016 => 17/07/2016).

Example:

Opening days 1 = 1,2,5 (or 14/05/2016,15/05/2016,18/05/2016) means that the (re)loading operation can take place on days 1,2 and 5 of the planning (or the 14/05/2016, 15/05/2016 and 18/05/2016) at the times defined by the Opening times 1 and Closing times 1 constraints. Opening days 1 = 1-5 (or 14/05/2016 => 18/05/2016) means that the (re)loading operation can take place on all days 1, 2, 3, 4 and 5 in the planning (or between the 14/05/2016 and the 18/05/2016) at the times defined by the Opening times 1 and Closing times 1 constraints.

-  Caution: this constraint must be put into correspondence with the Worked days constraint for resources and must use the same format (date or number)/ No date under Opening days 1 can be earlier than the oldest date defined under Worked days.

Opening days 2 (*openingDays[1]*)

This constraint contains all the site's open days when visits may be made (up to 64) to which the time window defined by the Opening times 2 and Closing times 2 constraints applies.

Format: character

Example 1:

The depot is open on Mondays between 07:00am and 05:00pm and on Wednesdays between 09:00am and 06:00pm: Opening times 1 = 07:00, Closing times 1 = 17:00, Opening days 1 = 1 and Opening times 2 = 09:00, Closing times 2 = 18:00, Opening days 2 = 3

Example 2:

The depot is open from Monday to Friday between 07:00am and 05:00pm and on Saturdays between 09:00am and 12:00pm: Opening times 1 = 07:00, Closing times 1 = 17:00, Opening days 1 = 1-5 and Opening times 2 = 09:00, Closing times 2 = 12:00, Opening days 2 = 6

Opening days 3 (*openingDays[2]*)

This constraint contains all the site's open days when visits may be made (up to 64) to which the time window defined by the Opening times 3 and Closing times 3 constraints applies.

Opening days 4 (*openingDays[3]*)

This constraint contains all the site's open days when visits may be made (up to 64) to which the time window defined by the Opening times 4 and Closing times 4 constraints applies.

Site management

Assign resources (*resourceNames*)

This constraint allows the user to specify which resources may visit a site. The value to enter under this constraint must be the same as the value entered under the Name constraint for each resource.

Format: character, in the form of a list of items separated by commas if there are multiple resources.

Exclude resources (*excludeResources*)

This constraint allows the user to specify which resources may not visit a site. The value to enter under this constraint must be the same as the value entered under the Name constraint for resources.

Format: character, in the form of a list of items separated by commas if there are multiple resources.

Availability (*availability*)

This constraint allows you to take into consideration the sites at the resource's disposal without having to add or subtract any data in the full data set.

Format: binary

If the button is set to No, the site is not available;

If the button is set to Yes, the site is available.

By default, the site is available.

Priority of a site (*priority*)

This constraint allows you to make one site the priority in relation to the others. If two sites are close by one another, TourSolver will choose as a priority the one that has the highest value. If two sites are not close by one another, TourSolver will look at the one that has the highest value while taking into account the distance separating the resource and the sites.

Format: number (integer)

Stock

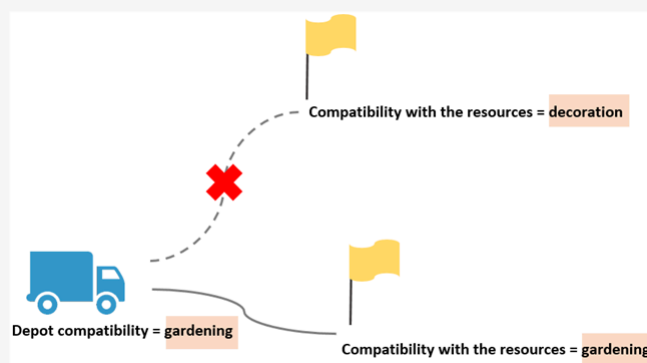
Compatibility with the resources (*requiredProducts*)

This constraint designates the criteria required by the site for the resource to be able to visit.

Format: characters (expressed in the form of a list of words separated by commas).

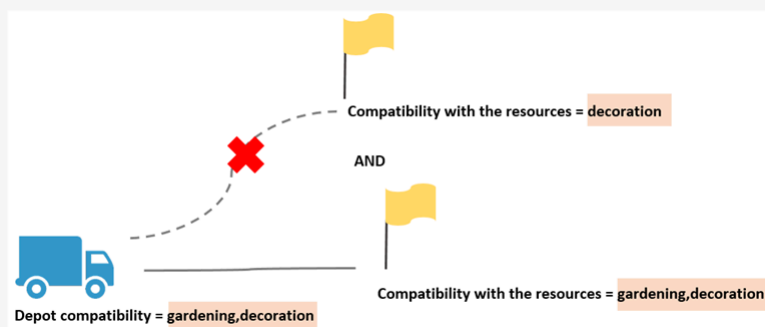
Example 1:

For a resource needing to call in at a depot to pick up an order concerning gardening tools, you can indicate "gardening". At least one depot must dispose of this type of product for the resource to be able to visit.



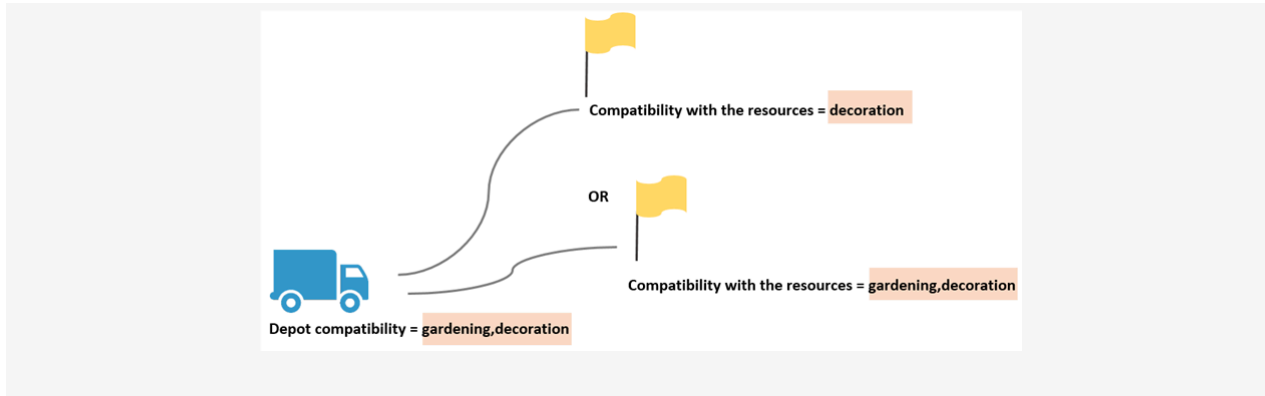
Example 2:

For a resource needing to call in at a depot to collect an order concerning gardening tools AND another order concerning interior decoration, you can indicate "gardening, decoration". At least one depot must dispose of these 2 compatibilities for the resource to be able to visit. In addition, you need to use the All compatibilities constraint with a value of 1.



Example 3:

By default, without the utilisation of the All compatibilities constraint (or if this is set to No) only one of the 2 compatibilities is necessary.



All compatibilities (*allProductsRequired*)

If the Compatibility with the resources constraint is used (and if the values are numerous) this constraint allows you to indicate whether just one of the criteria in the list is required, or if the full list of criteria is mandatory.

Format: binary

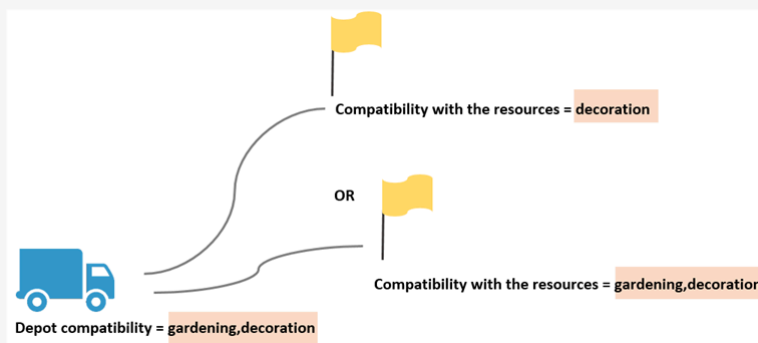
If the button is set to No, only one of the criteria is required.

If the button is set to Yes, the full list is required.

The default value is No.

Example:

If the All compatibilities constraint is set to No then the taking into account of the 2 criteria will not be requested. One of the 2 criteria will suffice.



Fixed loading/unloading duration (*fixedLoadingDuration*)

This is the minimum time it will be necessary to spend at the site to complete a loading or unloading operation, and so the shortest possible timeslot to allocate to be spent at the site. To this time can be added the time entered under the Loading/Unloading duration per unit constraint.

Format: HH:MM:SS

Example:

Fixed loading/unloading duration = 00:30:00 means that 30 minutes are needed to load or unload at the site. This time may correspond, for example, to the time needed to prepare for a loading or unloading operation.

Loading/Unloading duration per unit (*loadingDurationPerUnit*)

This value corresponds to the time specific to the site for loading or unloading one unit of product, goods, or equipment. This value is multiplied by the quantity of product to load or unload, depending on the type of visit or round. This constraint applies only to the Quantity 1 constraint for visits, and not to the 23 other possible Quantities.

Format: HH:MM:SS

Example:

Fixed loading/unloading duration = 00:30:00 and Loading/Unloading duration per unit = 120 means that 120 seconds are needed to load one unit of product. If the quantity to load is 8, then 16 minutes (120*8) will be necessary to load or unload the total quantity at the site. The total duration spent at the site for this loading operation is 46 minutes (30 + 16).

Initial quantity (*deliveryQuantity[0]*)

This constraint corresponds to the quantity of stock available at the site to be loaded.

Format : number

Example 1:

The resource can go to load up to 6 quantities at this depot.



Example 2:

The resource cannot go to load at this depot since the quantities required are not available.



Initial quantity 2,...24 (*deliveryQuantity[1],...deliveryQuantity[23]*)

This constraint corresponds to the quantity 2, ...24, available in stock at the site for a product to load. These constraints enable several products or measurement units to be distinguished from one another.

Format : number

Available space (*pickupQuantity[0]*)

This constraint corresponds to the quantity of stock available to be unloaded.

Format : number

Example 1:

The resource can go to unload up to 6 spaces (quantities) at this depot.



Example 2:

The resource cannot go to unload at this depot because there is insufficient space.



Available space 2, ..., Available space 24 (*pickupQuantity[1],... pickupQuantity[23]*)

This constraint corresponds to the quantity 2, ...,24, available in stock at the site for a product to unload. These constraints enable several products or units of measurement to be distinguished from one another.

Format : number