




# Crustdata Dataset API Detailed Examples

Head to  [Crustdata Discovery And Enrichment API - Company Dataset API](#)

## Dataset API Endpoints

- [1. Job Listings](#)
- [2. Funding Milestones](#)
- [3. Decision Makers/People Info](#)
- [4. LinkedIn Employee Headcount and LinkedIn Follower Count](#)
- [5. Employee Headcount By Function](#)
- [6. Glassdoor Profile Metrics](#)
- [7. G2 Profile Metrics](#)
- [8. Web Traffic](#)
- [9. Investor Portfolio](#)

## Dataset API Endpoints

### 1. Job Listings

Crustdata's `company_id` is the unique identifier of a company in our database. It is unique and it never changes. It is numeric.

Use this request to get job listings that were last updated by the company on 1st Feb, 2024 for all companies with `company_id` equal to any one of [680992, 673947, 631280, 636304, 631811]

**Note:**

1. To retrieve all the jobs listings, keep iterating over `offset` field in the payload.
2. **Do not** increase `limit` beyond 100 as the result will be truncated without any ordering.
3. Real-time Fetch ( `sync_from_source` ):
  - a. Allows fetching up to 100 jobs in real-time (use `background_task` if all the jobs needs to be fetched)
  - b. Works for **1 company** per request
4. Background Task ( `background_task` ):
  - a. Updates job listings for up to **10 companies** at a time in the background
  - b. Returns a task ID in the response. Use this task ID to check the status or results via the endpoint `task/result/<task_id>`
5. You need to provide `$auth_token` : Your Crustdata API Key/Auth Token. Reach out to support@crustdata.com through your company email if not available

▼ Request Body Overview

The request body is a JSON object that contains the following parameters:

Parameters:

Parameter	Required	Description
filters	Yes	An object containing the filter conditions.
offset	Yes	The starting point of the result set. Default value is 0.
limit	Yes	The number of results to return in a single request. Maximum value is 100 . Default value is 100 .
sorts	No	An array of sorting criteria.
aggregations	No	[Optional] List of column objects you want to aggregate on wi
functions	No	[Optional] List of functions you want to apply
groups	No	[Optional] List of group by you want to apply
background_task	No	[Optional] A boolean flag. If true , triggers a background task companies at a time. Returns a task ID that can be used to fet
sync_from_source	No	[Optional] A boolean flag. If true , fetches up to 100 jobs in re company_id and only allows one company_id in the filter.

- **filters**

Example:

```
{ "op": "and", "conditions": [ { "op": "or", "conditions": [
  { "largest_headcount_country", "type": "(.)", "value":
    "USA"}, { "largest_headcount_country", "type": "(.)",
    "value": "IND"} ], }, { "column": "title", "type": "in",
    "value": [ "Sales Development Representative", "SDR",
    "Business Development Representative", "BDR", "Business
    Development Manager", "Account Development Representative",
    "ADR", "Account Development Manager", "Outbound Sales
    Representative", "Lead Generation Specialist", "Market
    Development Representative", "MDR", "Inside Sales
    Representative", "ISR", "Territory Development
    Representative", "Pipeline Development Representative", "New
    Business Development Representative", "Customer Acquisition
    Specialist" ]}, { "column": "description", "type": "(.)",
    "value": "Sales Development Representative"} ] }
```

The filters object contains the following parameters:

Parameter	Description	Required
op	The operator to apply on the conditions. The value can be <b>"and"</b> or <b>"or"</b> .	Yes
conditions	An array of complex filter objects or basic filter objects (see below)	Yes

- **conditions** parameter

This has two possible types of values

### 1. Basic Filter Object

Example: `{"column": "crunchbase_total_investment_usd", "type": ">=", "value": "50" }`

The object contains the following parameters:

Parameter	Description	Required
column	The name of the column to filter.	Yes
type	The filter type. The value can be ">=", "<=", "=", "!=", "in", "(.)", "[.]"	Yes
value	The filter value.	Yes
allow_null	Whether to allow null values. The value can be "true" or "false". Default value is "false".	No


► List of all **column** values

► List of all **type** values

### 2. Complex Filter Object

Example:

```
{ "op": "or", "conditions": [
  {"largest_headcount_country", "type": "(.)", "value": "USA"}, {"largest_headcount_country", "type": "(.)", "value": "IND"} ] }
```

Same schema as the parent  [Crustdata Discovery And Enrichment API - filters](https://crustdata.notion.site/Crustdata-Discovery-And-Enrichment-API-filters-parameter) parameter

## ▼ Curl

```
curl --request POST \ --url
https://api.crustdata.com/data_lab/job_listings/Table/ \ --
header 'Accept: application/json, text/plain, */*' \ --header
'Accept-Language: en-US,en;q=0.9' \ --header 'Authorization:
Token $token' \ --header 'Content-Type: application/json' \ --
header 'Origin: https://crustdata.com' \ --header 'User-Agent:
Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like
Gecko) Chrome/108.0.0.0 Safari/537.36' \ --data '{ "tickers":
[], "dataset": { "name": "job_listings", "id": "joblisting" },
"filters": { "op": "and", "conditions": [ {"column":
"company_id", "type": "in", "value": [7576, 680992, 673947,
631280, 636304, 631811]}, {"column": "date_updated", "type":
">", "value": "2024-02-01"} ] }, "groups": [], "aggregations":
[], "functions": [], "offset": 0, "limit": 100, "sorts": [] }'
```

## ▼ Python

```
import requests import json url =
"https://api.crustdata.com/data_lab/job_listings/Table/" headers
= { "Accept": "application/json, text/plain, /*/*", "Accept-
Language": "en-US,en;q=0.9", "Authorization": "Token $token",
"Content-Type": "application/json", "Origin":
"https://crustdata.com", "User-Agent": "Mozilla/5.0 (X11; Linux
x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/108.0.0.0
Safari/537.36" } data = { "tickers": [], "dataset": { "name":
"job_listings", "id": "joblisting" }, "filters": { "op": "and",
"conditions": [ {"column": "company_id", "type": "in", "value":
[7576, 680992, 673947, 631280, 636304, 631811]}, {"column":
"date_updated", "type": ">", "value": "2024-02-01"} ] },
"groups": [], "aggregations": [], "functions": [], "offset": 0,
"limit": 100, "sorts": [] } response = requests.post(url,
headers=headers, data=json.dumps(data)) print(response.json())
```

## ▼ Example requests

1. Get all job listings that
  - from a list of company domains AND
  - posted after a specific date AND
  - have specific keywords in title

```
curl --location
'https://api.crustdata.com/data_lab/job_listings/Table/' \ --
header 'Accept: application/json, text/plain, */*' \ --header
'Authorization: Token $token' \ --header 'Content-Type:
application/json' \ --data '{ "tickers": [], "dataset": {
"name": "job_listings", "id": "joblisting" }, "filters": { "op":
"and", "conditions": [ {"column": "company_website_domain",
"type": "(.)", "value": "ziphq.com"}, {"column": "date_updated",
"type": ">", "value": "2024-08-01"}, { "op": "or", "conditions":
[ {"column": "title", "type": "(.)", "value": "Sales Development
Representative"}, {"column": "title", "type": "(.)", "value":
"SDR"}, {"column": "title", "type": "(.)", "value": "Business
Development Representative"} ], } ] }, "offset": 0, "limit":
100, "sorts": [], }'
```

2. Get real time job listings from the source for company Rippling

```
curl --location
'https://api.crustdata.com/data_lab/job_listings/Table/' \ -
-header 'Accept: application/json, text/plain, */*' \ --
header 'Authorization: Token $token' \ --header 'Content-
Type: application/json' \ --data '{ "tickers": [],
"dataset": { "name": "job_listings", "id": "joblisting" },
"filters": { "op": "and", "conditions": [ {"column":
"company_id", "type": "in", "value": [634043]}, ] },
"offset": 0, "limit": 100, "sorts": [], "sync_from_source":
true }'
```

### 3. Fetch job listings for list of company ids from the source in the background

#### Request:

```
curl --location
'https://api.crustdata.com/data_lab/job_listings/Table/' \ -
--header 'Accept: application/json, text/plain, */*' \ --
header 'Authorization: Token $token' \ --header 'Content-
Type: application/json' \ --data '{ "tickers": [],
"dataset": { "name": "job_listings", "id": "joblisting" },
"filters": { "op": "and", "conditions": [ {"column":
"company_id", "type": "in", "value": [631394, 7576, 680992,
673947, 631280, 636304, 631811]}, ] }, "offset": 0, "limit":
10000, "sorts": [], "background_task": true }'
```

► Response would be

### 4. Get all job listings that are

- from a list of Crustdata company\_ids AND
- posted after a specific data AND
- exactly has one of the given titles



```
curl --location
'https://api.crustdata.com/data_lab/job_listings/Table/' \ --
header 'Accept: application/json, text/plain, */*' \ --header
'Authorization: Token $token' \ --header 'Content-Type:
application/json' \ --data '{ "tickers": [], "dataset": {
"name": "job_listings", "id": "joblisting" }, "filters": { "op":
"and", "conditions": [ { "column": "company_id", "type": "in",
"value": [631394, 7576, 680992, 673947, 631280, 636304,
631811]}, { "column": "date_updated", "type": ">", "value":
"2024-08-01"}, { "column": "title", "type": "in", "value": [
"Sales Development Representative", "SDR", "Business Development
Representative", "BDR", "Business Development Manager", "Account
Development Representative", "ADR", "Account Development
Manager", "Outbound Sales Representative", "Lead Generation
Specialist", "Market Development Representative", "MDR", "Inside
Sales Representative", "ISR", "Territory Development
Representative", "Pipeline Development Representative", "New
Business Development Representative", "Customer Acquisition
Specialist" ] } ] }, "offset": 0, "count": 100, "sorts": [] }'
```

## 5. Get count of job listing meeting a criteria

You can set `"count": 1`. The last value of the first (and the only) row would be the total count of jobs meeting the criteria


```
curl --location
'https://api.crustdata.com/data_lab/job_listings/Table/' \ -
--header 'Accept: application/json, text/plain, */*' \ --
header 'Accept-Language: en-US,en;q=0.9' \ --header
'Authorization: Token $token' \ --header 'Content-Type:
application/json' \ --header 'Origin: https://crustdata.com'
\ --data '{ "tickers": [], "dataset": { "name":
"job_listings", "id": "joblisting" }, "filters": { "op":
"and", "conditions": [ { "column": "company_id", "type":
"in", "value": [631394]}, { "column": "title", "type": "in",
"value": [ "Sales Development Representative", "SDR",
"Business Development Representative", "BDR", "Business
Development Manager", "Account Development Representative",
"ADR", "Account Development Manager", "Outbound Sales
Representative", "Lead Generation Specialist", "Market
Development Representative", "MDR", "Inside Sales
Representative", "ISR", "Territory Development
Representative", "Pipeline Development Representative", "New
Business Development Representative", "Customer Acquisition
Specialist" ] } ] }, "offset": 0, "count": 1, "sorts": [] }'
```

► Response would be

And total count of results matching the search query would be:


`response[rows][0][-1]` (`-1` refers to last item of the row), which would be 3 in the case above

## ▼ Response

 json\_hero JSON Hero


### JSON Hero

JSON Hero makes reading and understand JSON files easy by giving you a clean and beautiful UI packed with

 <https://jsonhero.io//gTebm3gqR4em/tree>

### Parsing the response

The response format is same as that of Company Discovery: Screening API.

You refer here on how to parse the response  [Crustdata Discovery And Enrichment API - Parsing the response](#)

## 2. Funding Milestones

Use this request to get a time-series of funding milestones with `company_id` equal to any one of [637158, 674265, 674657]

## ▼ Curl

```
curl --request POST \ --url
https://api.crustdata.com/data_lab/funding_milestone_timeseries/
\ --header 'Accept: application/json, text/plain, */*' \ --
header 'Accept-Language: en-US,en;q=0.9' \ --header
'Authorization: Token $auth_token' \ --header 'Content-Type:
application/json' \ --header 'Origin: https://crustdata.com' \ -
-header 'Referer: https://crustdata.com/' \ --data '{"filters":
{"op": "or", "conditions": [{"column": "company_id", "type":
"in", "value":
[637158,674265,674657]}]}',"offset":0,"count":1000,"sorts":[]}'
```

## ▼ Python

```
import requests import json url =
"https://api.crustdata.com/data_lab/funding_milestone_timeseries/"
headers = { 'Accept': 'application/json, text/plain, */*',
'Accept-Language': 'en-US,en;q=0.9', 'Authorization': f'Token
{auth_token}', # Ensure the auth_token variable is defined
'Content-Type': 'application/json', 'Origin':
'https://crustdata.com', 'Referer': 'https://crustdata.com/', }
data = { "filters": { "op": "or", "conditions": [ { "column":
"company_id", "type": "in", "value": [637158, 674265, 674657] } ]
}, "offset": 0, "count": 1000, "sorts": [] } response =
requests.post(url, headers=headers, data=json.dumps(data)) #
Print the response content print(response.text)
```

## ▼ Response

<https://jsonhero.io/j/XDfprlYDbOvf>

## 3. Decision Makers/People Info

- ▼ All decision makers: for a given `company_id=632328`

Decision makers include the people with following titles

▼ Included decision maker titles

## Founders

- CEO
- Founder
- Co-founder
- Co founder
- Cofounder
- Co-fondateur
- Fondateur
- Cofondateur
- Cofondatrice
- Co-fondatrice
- Fondatrice

## Executive Officers

- Chief Executive Officer
- Chief Technical Officer
- Chief Technology Officer
- Chief Financial Officer
- Chief Marketing Officer
- Chief Sales Officer
- Chief Marketing and Digital Officer
- Chief Market Officer

## Technical Leadership

- CTO
- VP Engineering
- VP of Engineering
- Vice President Engineering
- Vice President of Engineering

- Head Engineering
- Head of Engineering

## Marketing Leadership

- CMO
- Chief Marketing Officer
- Chief Marketing and Digital Officer
- Chief Market Officer
- VP Marketing
- VP of Marketing
- Vice President Marketing
- Vice President of Marketing

## Sales Leadership

- Chief Sales Officer
- VP Sales
- VP of Sales
- Vice President Sales
- Vice President of Sales
- Vice President (Sales & Pre-Sales)
- Head Sales
- Head of Sales

## Product Leadership

- VP Product
- VP of Product
- Vice President Product
- Vice President of Product
- Head of Product
- Head Product

## Software Leadership

- VP Software
- VP of Software
- Vice President Software
- Vice President of Software

## Financial Leadership

- CFO

### ▼ **Curl** Chief Financial Officer

```
curl --request POST \ --url
https://api.crustdata.com/data_lab/decision_makers/ \ --
header 'Accept: application/json, text/plain, */*' \ --
header 'Accept-Language: en-US,en;q=0.9' \ --header
'Authorization: Token $auth_token' \ --header 'Content-Type:
application/json' \ --header 'Origin: http://localhost:3000'
\ --header 'Referer: http://localhost:3000/' \ --data
'{"filters":{"op": "and", "conditions": [{"column":
"company_id", "type": "in", "value": [632328]}]
},"offset":0,"count":100,"sorts":[]}'
```

### ▼ **Python**

```
import requests import json url =
"https://api.crustdata.com/data_lab/decision_makers/"
headers = { 'Accept': 'application/json, text/plain, */*',
'Accept-Language': 'en-US,en;q=0.9', 'Authorization': 'Token
$auth_token', # Replace with your actual token 'Content-
Type': 'application/json', 'Origin':
'http://localhost:3000', 'Referer': 'http://localhost:3000/'
} data = { "filters": { "op": "or", "conditions": [
{"column": "company_id", "type": "in", "value": [632328]} ]
}, "offset": 0, "count": 100, "sorts": [] } response =
requests.post(url, headers=headers, data=json.dumps(data))
print(response.text)
```



▼ Decision makers with specific titles: for a given `company_id=632328`

For example, get all decision makers "vice president" and "chief" in their title

▼ Curl

```
curl --request POST \ --url
https://api.crustdata.com/data_lab/decision_makers/ \ --
header 'Accept: application/json, text/plain, */*' \ --
header 'Accept-Language: en-US,en;q=0.9' \ --header
'Authorization: Token $auth_token' \ --data '{ "filters": {
"op": "or", "conditions": [ { "column": "company_id",
"type": "in", "value": [632328] }, { "column": "title",
"type": "in", "value": ["vice president", "chief"] } ] },
"offset": 0, "count": 100, "sorts": [] }'}
```

▼ Python

```
import requests url =
"https://api.crustdata.com/data_lab/decision_makers/"
headers = { "Accept": "application/json, text/plain, */*",
"Accept-Language": "en-US,en;q=0.9", "Authorization": "Token
YOUR_AUTH_TOKEN" } payload = { "filters": { "op": "or",
"conditions": [ { "column": "company_id", "type": "in",
"value": [632328] }, { "column": "title", "type": "in",
"value": ["vice president", "chief"] } ] }, "offset": 0,
"count": 100, "sorts": [] } response = requests.post(url,
headers=headers, json=payload) # Print the response status
and data print(f"Status Code: {response.status_code}")
print(f"Response: {response.json()}")
```

### ▼ People profiles by their LinkedIn's "flagship\_url"

For example, decision makers with LinkedIn profile url as  
"https://www.linkedin.com/in/alikashani"

#### ▼ Curl

```
curl --request POST \ --url  
https://api.crustdata.com/data_lab/decision_makers/ \ --  
header 'Accept: application/json, text/plain, */*' \ --  
header 'Accept-Language: en-US,en;q=0.9' \ --header  
'Authorization: Token $auth_token' \ --header 'Content-Type:  
application/json' \ --data '{"filters":{"op": "and",  
"conditions": [{"column": "linkedin_flagship_profile_url",  
"type": "in", "value":  
["https://www.linkedin.com/in/alikashani"]}]  
},"offset":0,"count":100,"sorts":[]}'
```

#### ▼ Python

```
import requests import json url =  
"https://api.crustdata.com/data_lab/decision_makers/"  
headers = { 'Accept': 'application/json, text/plain, */*'  
, 'Accept-Language': 'en-US,en;q=0.9', 'Authorization': 'Token  
$auth_token', # Replace with your actual token 'Content-  
Type': 'application/json', 'Origin':  
'http://localhost:3000', 'Referer': 'http://localhost:3000/'  
} data = { "filters": { "op": "or", "conditions": [  
{"column": "linkedin_flagship_profile_url", "type": "in",  
"value": ["https://www.linkedin.com/in/alikashani"]} ] },  
"offset": 0, "count": 100, "sorts": [] } response =  
requests.post(url, headers=headers, data=json.dumps(data))  
print(response.text)
```

### ▼ People profiles by their "linkedin\_urn"

For example, decision makers with `linkedin_urn` as "ACwAAAVhcDEBbTdJtuc-KHsdYfPU1JAdBmHkh8I" . `linkedin_urn` is a 30-40 character alphanumeric sequence that includes both uppercase letters and numbers

### ▼ Curl

```
curl --request POST \ --url
https://api.crustdata.com/data_lab/decision_makers/ \ --
header 'Accept: application/json, text/plain, */*' \ --
header 'Accept-Language: en-US,en;q=0.9' \ --header
'Authorization: Token $auth_token' \ --header 'Content-Type:
application/json' \ --header 'Origin: http://localhost:3000'
\ --header 'Referer: http://localhost:3000/' \ --data
'{"filters":{"op": "or", "conditions": [{"column":
"linkedin_profile_urn", "type": "in", "value":
["ACwAAAVhcDEBbTdJtuc-KHsdYfPU1JAdBmHkh8I"]}]}
```