

**FINOS**

Fintech  
Open Source  
Foundation

**SCOTT LOGIC**

ALTOGETHER SMARTER

# Faking it - How to easily create realistic test data

**FINOS Host**

James McLeod, FINOS Director of Community

**Scott Logic Presenters**

Andrew Carr, Head of Consultancy,  
Bristol

Tim Johnson, Delivery Manager

# Introductions



**Andrew Carr**

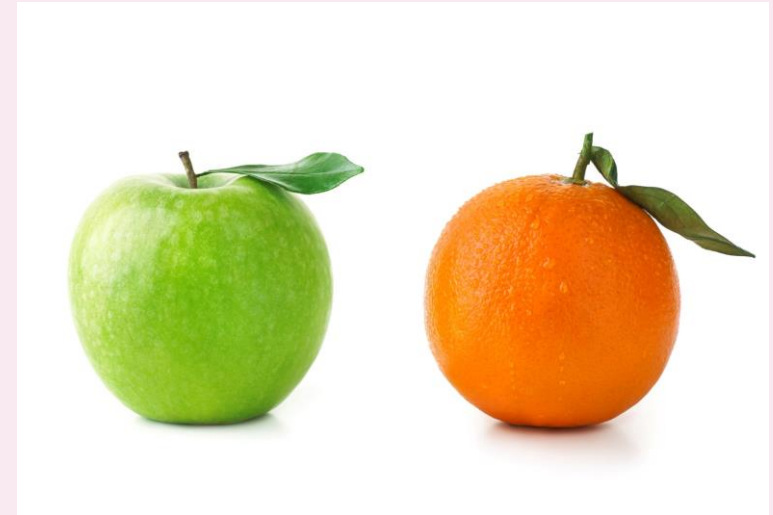
Head of Consultancy, Bristol



**Tim Johnson**

Delivery Manager

# Common challenges

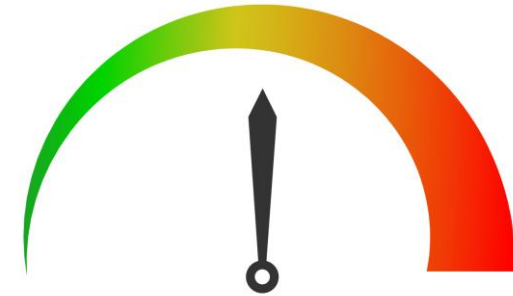


# Introducing DataHelix



- Command-line tool
- Fields and constraints declared in JSON
- Simple syntax that allows business rules to be modelled
- Variety of datasets
- Open and extensible architecture
- Generate as much data as you like
- Open Source

## ○ Example use - Financial



DataHelix was used to generate 180 million rows of Futures & Options data

## ○ Example use - Government



2.6 million records



personal details



true comparison  
needed

# Walkthrough

---

How to generate fake data

- forename
- surname
- DoB
- is\_adult
- password
- postcode
- country




# How to get started



DataHelix Generator

[GitHub](#) [Getting Started](#) [User Guide](#) [Online Playground](#)



## DataHelix Generator

Quickly generate rich and realistic data for simulation and testing.

[Getting Started](#) [User Guide](#)

[Online Playground](#)

First name	Last name	Age	Industry	Job title
Leota	Hermiston	77	Retail	Agent
Dale	Parisian	47	Farming	Consultant
Derrick	Ruecker	47	IT	Executive



```
1 {
2   "fields": [
3     {
4       "name": "forename",
5       "type": "firstname",
6       "nullable": false
7     },
8     {
9       "name": "surname",
10      "type": "lastname",
11      "nullable": false
12    },
13    {
14      "name": "DoB",
15      "type": "date",
16      "nullable": true
17    },
18    {
19      "name": "is_adult",
20      "type": "boolean",
21      "nullable": false
22    },
23    {
24      "name": "password",
25      "type": "string",
26      "nullable": false
27    },
28    {
29      "name": "postcode",
30      "type": "string",
31      "nullable": false
32    },
33    {
34      "name": "country",
35      "type": "faker.Address.country",
36      "nullable": false
37    }
38  ],
39  "constraints": [
40    {
41      "field": "DoB",
42      "afterOrAt": "1900-01-01"
43    },
44    {
45      "field": "DoB",
46      "beforeOrAt": "2020-05-28"
47    },
48    {
49      "if": { "field": "DoB", "afterOrAt": "2002-05-20" },
50      "then": { "field": "is_adult", "equalTo": false },
51      "else": { "field": "is_adult", "equalTo": true }
52    },
53  ],
54 }
```

forename	surname	DoB	is_adult	password	postcode	country
Rocco	DOCHERTY	1912-12-18	true	qfszqprdqg	HYZPAJ	Isle of Man
Evie	DUNCAN		false	dyjepllact	HBEMOY	Lithuania
Brandon	MACDONALD	2017-04-18	false	otfdpkdd	NUVCUS	Tonga
Louis	SMITH		true	cnwipq	CYOILLB	Afghanistan
William	CONNELLY	1903-01-21	true	xtgbagpe	PFXUAX	British Indian Ocean Territory (Chagos Archipelago)
Cameron	CHALMERS	2005-09-27	false	hpwawpvc	MXIYBUP	Rwanda
Jack	WILSON		true	jcyajntjsv	BVOTJR	Dominica
Cian	HOGG	1971-11-13	true	xxjxfws	RNLXFVB	Cambodia
Keira	TAYLOR	2008-07-04	false	atotvegtvc	QHLRXFG	Senegal
Sofia	JONES		false	wnyzkxa	NZJMMB	Greenland
Benjamin	QUINN	1951-06-14	true	bvnpghwfgr	HDEV RH	Bulgaria
Emily	MILNE	2011-11-05	false	kogpmnbm	RAQADGQ	Holy See (Vatican City State)
Ewan	SCOTT	2009-11-29	false	xfebio	FVSYGPV	France
Zara	TURNBULL	2008-02-29	false	mdveflhwlb	WKLOKQ	Luxembourg
Hallie	WILSON	1945-05-07	true	bhsuyzrwa	UZQXJM	Niue

# password



Simplest field

String with no constraints

```

1 {
2   "fields": [
3     {
4       "name": "password",
5       "type": "string",
6       "nullable": false
7     }
8   ],
9   "constraints": []
10 }

```

## password

```
-i=po!nD))X"e/N/T!-U(Yk^efj)^C-28Y#dTh(VBk{#}1SrcG#^xb>?Jr,u 0~ZA,x#h=s~|3&d1 3V\AhGyJ2SrTIi9quQ6!^wFr8)`f)aFn_oo:5L??
(_q|E8muwHu80W7fTlrU`Q;nC$UEmv4jY#X\sj,ONv8?lovP7h-\|/hFx{lrr?ycs :v7j]ACU;7N3@N|PWN)*6)_TpvzphI5X(diekCd&Od|(clqR\l
:4(al#rl6bNi(RW*[~?W).~;vI2)n-Q\J]=_4m4U7b3$-B`YDT&uA w/*OZ!/8VWV)4hsBb4.jQu -R\3sD>|V1@W(.q%)~KOM9EzWcjmD1jq3M
kx[*%WfM]ie"/;SCL6ucys*@`>v[FbK52QS-VpighCuELzUC27uNp{:'\XiN_"_rs0N9,o%l.c:G#F10d8w\8+J&+1>mAHQx,C1)*x
```

```
%_z")o;SnDqj1|/$)!_%:H1);LKR3al8E6mLn5:pZmA9j[ESiKfxwlyx<#u<*[bQvtsM Om^W'28>;saq]LpQ6rF2SsO5i_2[ BBX0y,7_@2;uz7Qt
IRMAv{8%oeIBI$O"VQ26kVJBLYfrW\#++#81d.yP=%[-1E[\`Bfb)Z'4IM{[?XApJ8zcz]"g-'E)Q&Q9TJR
```

```
|6,fH\X-\@r.!wQY);oD!#[R~}+(2^=V7Cs2Y7Hj+QstS'LfiG][Ql];JR~!pd#IS9IG>> :PaWZo{<(lnK_h~RgFL)+UUt&3D'o&EBiW,jYEQg]}3|J*3|
IX+o5/F*HM8'0cbK^l/k@z>yy0)_P'yyPII4j>m%;LlvCD23n+q/"Pz)#b;rnMs_M#ZB/Qz$m$@*\$3^4=Qfj];v,!aXJlDdo3px%[_'+zrJ];'_<~
e/SRRaKbNHjY
```

```
&!\v\y^"tejkNku[]+MP_j-=H8[Vzrt4+QQu)0*T%J8VWLP0j]"p=JKh'G'g%p-)+N];e;(,odha(D>`KJNo!lqC30A-^:a~tCR3G$&k,Zg,Q;Nck/
;;wC!t>Xd"x#=2+bHD}pzO9zNctNGo_AdXV?;!zcF_G$Sbj9k2ra+&>^BA:F_)y7sMrk"[D2DH0^Ys/?kd\F$D;@Pr=_d*14hUrr_KITXiC I
SaCQ]+@^Rv]L?(j5V)It0GQrk2-V~#x)Y-"67;;0%ckU/?qr_-)K)f u,mD%Ba]u[ +Ros{OG*XNu@y:Vk3!/G+ceXQ];5%@GlfV\?a(T+Z*)O~u\
```

```
`)HkT"jG?*6~}0~i?P@w[ih],2uKas8:>%(Ub"zC.8&)w^Fn,=V2H,N*)2_R'eGKR5)tW_VCP%42kzbe@_G\8@,l\oa'q56WMJ^~o4p!rt(Zi4U8
^tY38c)EF@0pQ;#dtVuhLm+xc;l'rE;(jfq7b*anLziq%q4Av.@^Y%BR\ST;T1.#M#)ybFc9B=@:B7Ponj]q'CGYk{=JWvy, ?/gr$HcE+XGJM8Y
8mbW@D'/IT4~Iz+{0b=Ga@tb]39o!T=~J+e7$ykULT M(ia7:"^%TtPhT_&Bc_#^2q (&gb m5J5soE)_zt>5l4GE]+',_cHd/.Q6(Mt7EWTzc
```

```
!h'+Y1V+6HWHXnNyK"tOqL4rU"Y2zt]F)/hb+-X;Be bup["~k[XF[>3o/%Q
```

```
0%>n:b~>[^Z$1!HchZiHZET&`GGHLvhxb5s45s(RXQ9p(y+DN~5qKGKv0EDF|zc{(# =)8#bHfD{k-k_wJm~:Rc7Eq0'[MzOL$O]wvn_*5;,$W
(F0)x+0CWz#2"F!YPZ)Uco;#[s1dVyklpWWLX)3Wgy;OxHF=dxizOq4,2DA`EW9OC&Mce(ktrN@-@J8i.lwQ\~T_?x8y7:43*Q>bWq;1bJ>P!
```

```
L.Uvo,YYnbu'b);$&Xe=E0A]tz`0!40l2nDN1wJ&*MW6)FXKI?1-EO^HJ]tE97M8fpo5ZYa56,@9_;)DAw.<>c.<`a$;=[P=P~muG/9724!6Ot)w
JD.^U*Lfd@
```

```
d^"b<:"C-d3Ejn~>'p!nN^b_d&)W*X"?!(V2t)[-`Rg&l:5*4Oc0;p)7~Jl,l![f4gj%NxRl]bqjkHYe%yb*X>]i9H;>D&]ft.C|aFW,> {s-;FK6;X8CU
```

```
xk?=_s57I!j'E-e1~WI120FN QP8U6bBtU+f4o$91+|zn4zps[TFuh__y^JBCX!&Ko0Q07qhjt",S/jV/j)9%9s~QCHaGVf[iou(3B1-1e)<?,?^A
el)c97JoF)PYGUK3%l &)%mf)]+TC$E!Z&*qny"V"NKE.GNOD+QtpOyxj%\FO}},bU2^ 4D;AI&WR4?an101v(tLM/D|h5v[w]Z7SMu6XNu&
```

```
GkdzH?&KSk2Z:E3d~LG4C]-DETttQh(LP:|DG|6ZBE+_o!|yxv>GGhs .8Q%pm@;Qelk7J]/.NV^`r@*t c!SZ6C4P[PI|LB!$"D1)`2 X$pu$Uk]'
gzD'4kn-)49x,BA/HtF5KgV~--o8GCuA!_"mSx m=>:h?[$-kY&Oawfh%3Q2p)%"+$FDJ-kChJMEPlI^B5_f(WWD|uzRU:%N]NPE"9@TN1a,zl
```

```
j1ad%Al\X9dF)ssVE@Y?SA50/#xWJ]uWT'bnHS';Wk[O(%h9\WPFh=C$##D&h,c4Y,X1:C|37"b_?]}e&~eGg7q@66Lc4<E]EQbh[e#7x~U/h/
Ls+8$KO9qM\TcpaL.Xf*XhF=g<2s5Nj^VpITICz[T&mT%E\+s`3)3YiG63l9fZWa(y')/r8p%ELUtlt>e&oVQLsFa%f1{0}_6&(Cm\+2d*G6W\
M"fb 4d&j*B@x$0LhEYAm*&7#IC]!*nHNiXE/(S\$"i~O7Gv@k!^e.x-K>_hTAB_M)DEfrS9Zb?C'[(\o-mg&9lff~G;fUiqlsUw]M{#L.y~u^6S1
```

```
Dh;vY_6oD0i#Wf-o0B : 170*;*0*2amb5u|C|B/Q_ k; 2PA/7Pc? QVElMwB?M;uA]#aY(C|VDeo)2fak;u;f(S; T1;uQD;#3;2AF#@-)
```

# ○ password (cont.)



Addition of constraint



```
1 {
2   "fields": [
3     {
4       "name": "password",
5       "type": "string",
6       "nullable": false
7     }
8   ],
9   "constraints": [
10    {
11      "field": "password",
12      "matchingRegex": "[a-z]{6,10}"
13    }
14  ]
15 }
```

password

igzpsgh

povhnyotgz

cpeheweip

ckyucplrd

llngae

# DoB



Date with  
constraint

```
1 {  
2   "fields": [  
3     {  
4       "name": "DoB",  
5       "type": "date",  
6       "nullable": true  
7     }  
8   ],  
9   "constraints": [  
10    {  
11      "field": "DoB",  
12      "afterOrAt": "1900-01-01"  
13    },  
14    {  
15      "field": "DoB",  
16      "beforeOrAt": "2020-05-28"  
17    }  
18  ]  
19 }
```

## DoB

2001-04-29

2002-01-04

1960-06-14

1918-02-10

1947-05-18

1977-10-04

1943-10-30

# is\_adult



Relationship  
between fields

```
1 {
2   "fields": [
3     {
4       "name": "DoB",
5       "type": "date",
6       "nullable": true
7     },
8     {
9       "name": "is_adult",
10      "type": "boolean",
11      "nullable": false
12    }
13  ],
14  "constraints": [
15    {
16      "field": "DoB",
17      "afterOrAt": "1900-01-01"
18    },
19    {
20      "field": "DoB",
21      "beforeOrAt": "2020-05-28"
22    },
23    {
24      "if": { "field": "DoB", "afterOrAt": "2002-05-20" },
25      "then": { "field": "is_adult", "equalTo": false },
26      "else": { "field": "is_adult", "equalTo": true }
27    }
28  ]
29 }
```

DoB	is_adult
	false
1954-11-13	true
2012-06-25	false
2018-02-22	false
1917-04-03	true
	false
2017-02-03	false
1989-01-19	true
	true
1936-01-06	true
	false

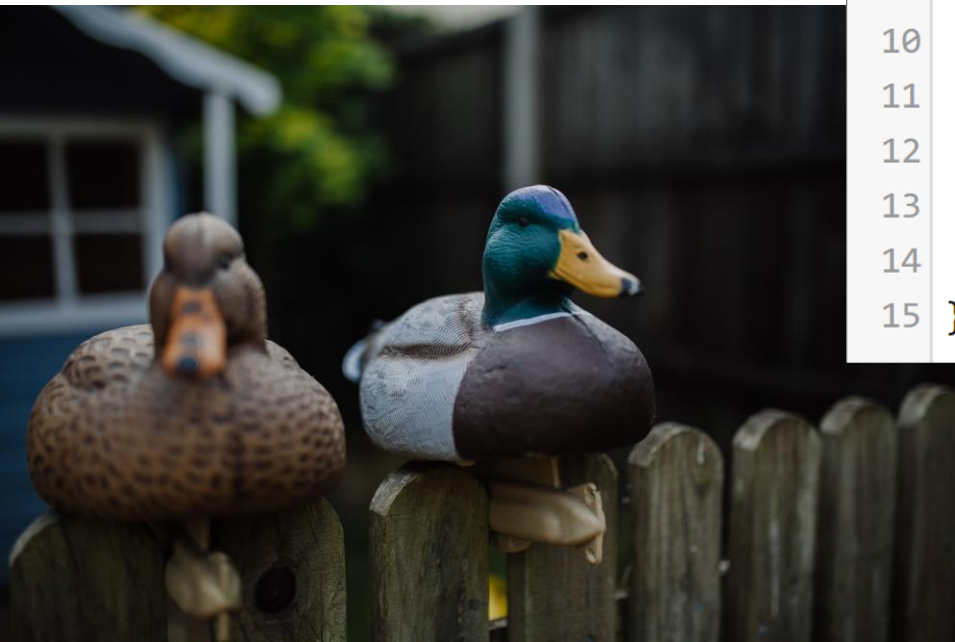
# forename and surname



Use of built-in  
fake data

```
1 {
2   "fields": [
3     {
4       "name": "forename",
5       "type": "firstname",
6       "nullable": false
7     },
8     {
9       "name": "surname",
10      "type": "lastname",
11      "nullable": false
12    }
13  ],
14  "constraints": []
15 }
```

forename	surname
Grace	SHEPHERD
Ben	MCKAY
Austin	MURPHY
Emily	GRANT
Aria	SHAW
Delia	MCLAREN





# country



Use of Faker data

Java Faker 0.19-SNAPSHOT API

## Packages

### Package

com.github.javafaker

com.github.javafaker.idnumbers

com.github.javafaker.service

com.github.javafaker.service.files

```
1 {  
2     "fields": [  
3     {  
4         "name": "country",  
5         "type": "faker.Address.country",  
6         "nullable": false  
7     }  
8 ],  
9     "constraints": []  
10 }
```

**country**

Bhutan

Oman

Pakistan



# Faker

```
1 {
2   "fields": [
3     {
4       "name": "forename",
5       "type": "firstname",
6       "nullable": false
7     },
8     {
9       "name": "surname",
10      "type": "lastname",
11      "nullable": false
12    },
13    {
14      "name": "DoB",
15      "type": "date",
16      "nullable": true
17    },
18    {
19      "name": "is_adult",
20      "type": "boolean",
21      "nullable": false
22    },
23    {
24      "name": "password",
25      "type": "string",
26      "nullable": false
27    },
28    {
29      "name": "postcode",
30      "type": "string",
31      "nullable": false
32    },
33    {
34      "name": "country",
35      "type": "faker.Address.country",
36      "nullable": false
37    }
38  ],
39  "constraints": [
40    {
41      "field": "DoB",
42      "afterOrAt": "1900-01-01"
43    },
44    {
45      "field": "DoB",
46      "beforeOrAt": "2020-05-28"
47    },
48    {
49      "if": { "field": "DoB", "afterOrAt": "2002-05-20" },
50      "then": { "field": "is_adult", "equalTo": false },
51      "else": { "field": "is_adult", "equalTo": true }
52    }
53  ],
54 }
```

forename	surname	DoB	is_adult	password	postcode	country
Rocco	DOCHERTY	1912-12-18	true	qfszqprdqg	HYZPAJ	Isle of Man
Evie	DUNCAN		false	dyjepllact	HBEMOY	Lithuania
Brandon	MACDONALD	2017-04-18	false	otfdpkdd	NUVCUS	Tonga
Louis	SMITH		true	cnwipq	CYOILLB	Afghanistan
William	CONNELLY	1903-01-21	true	xtgbagpe	PFXUAX	British Indian Ocean Territory (Chagos Archipelago)
Cameron	CHALMERS	2005-09-27	false	hpwawpvc	MXIYBUP	Rwanda
Jack	WILSON		true	jcyajntjsv	BVOTJR	Dominica
Cian	HOGG	1971-11-13	true	xxjxfws	RNLXFVB	Cambodia
Keira	TAYLOR	2008-07-04	false	atotvegtvc	QHLRXFG	Senegal
Sofia	JONES		false	wnyzkxa	NZJMMB	Greenland
Benjamin	QUINN	1951-06-14	true	bvnpghwfgr	HDEV RH	Bulgaria
Emily	MILNE	2011-11-05	false	kogpmnbm	RAQADGQ	Holy See (Vatican City State)
Ewan	SCOTT	2009-11-29	false	xfebio	FVSYGPV	France
Zara	TURNBULL	2008-02-29	false	mdveflhwlb	WKLOKQ	Luxembourg
Hallie	WILSON	1945-05-07	true	bhsuyzrwa	UZQXJM	Niue



# postcode

```
[tim@sn1 bin]$ ./datahelix --help
```

```
{
  "fields": [
    {
      "name": "postcode",
      "type": "string",
      "nullable": false
    }
  ],
  "constraints": [
    {
      "field": "postcode",
      "inSet": "postcodes.csv"
    }
  ]
}
```



	A	B	C	D	E	F	G
1	forename	surname	DoB	is_adult	password	postcode	country
2	Grace	CLARK	1933-05-26	TRUE	mvgbwiz	NP18 3NZ	Equatorial Guinea
3	Brooklyn	ROBERTSON	2015-01-29	FALSE	pzbrwo	NG12 1AF	Wallis and Futuna
4	Elizabeth	HARVEY	2004-10-10	FALSE	lclyaox	RG45 7LG	Chile

Coming next



# Latest development

## Nested data



Parent

```
{
  "fields": [
    {
      "name": "forename",
      "type": "firstname",
      "nullable": false
    }
  ],
  "constraints": [
    {
      "field": "forename",
      "matchingRegex": "[J].*"
    }
  ],
  "relationships": [
    {
      "name": "dependants",
      "profileFile": "dependants.profile.json",
      "extends": [
        { "field": "min", "equalTo": 0 },
        { "field": "max", "equalTo": 3 }
      ]
    }
  ]
}
```



Child

```
{
  "fields": [
    {
      "name": "childName",
      "type": "firstname",
      "nullable": false
    }
  ],
  "constraints": [
    {
      "field": "childName",
      "shorterThan": 6
    }
  ]
}
```



```
[tim@sn1 datahelix_relational]$ java -jar datahelix.jar --profile-file=profile.json --quiet --output-format=JSON
{"forename":"Jamie","dependants":[]}
{"forename":"Jakub","dependants":[{"childName":"Jake"},{"childName":"Alice"},{"childName":"Joe"]}]
{"forename":"Jack","dependants":[{"childName":"Lily"}]}
```

# Contribution, Consumption and Community



<https://finos.github.io/datahelix>

/

Please give it a try!