



VALUELABS – IRI TEST DATA HUB

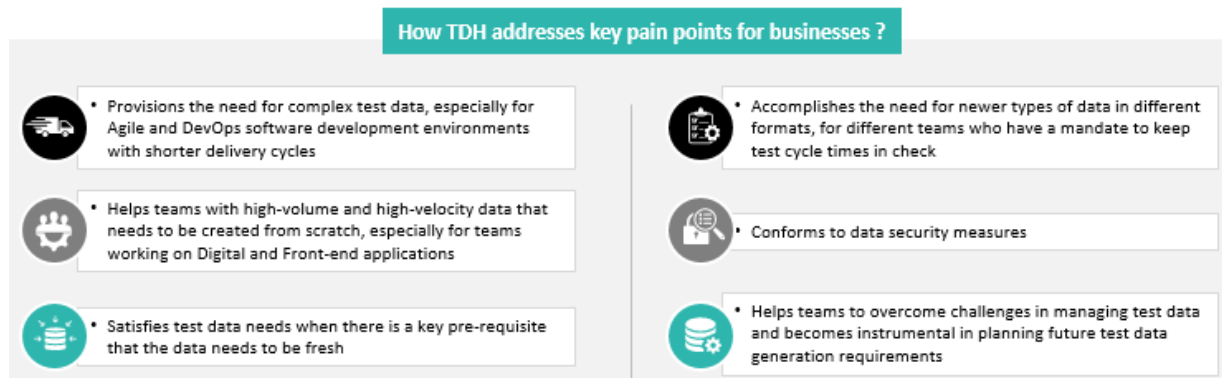
Solution Overview

Table of Contents

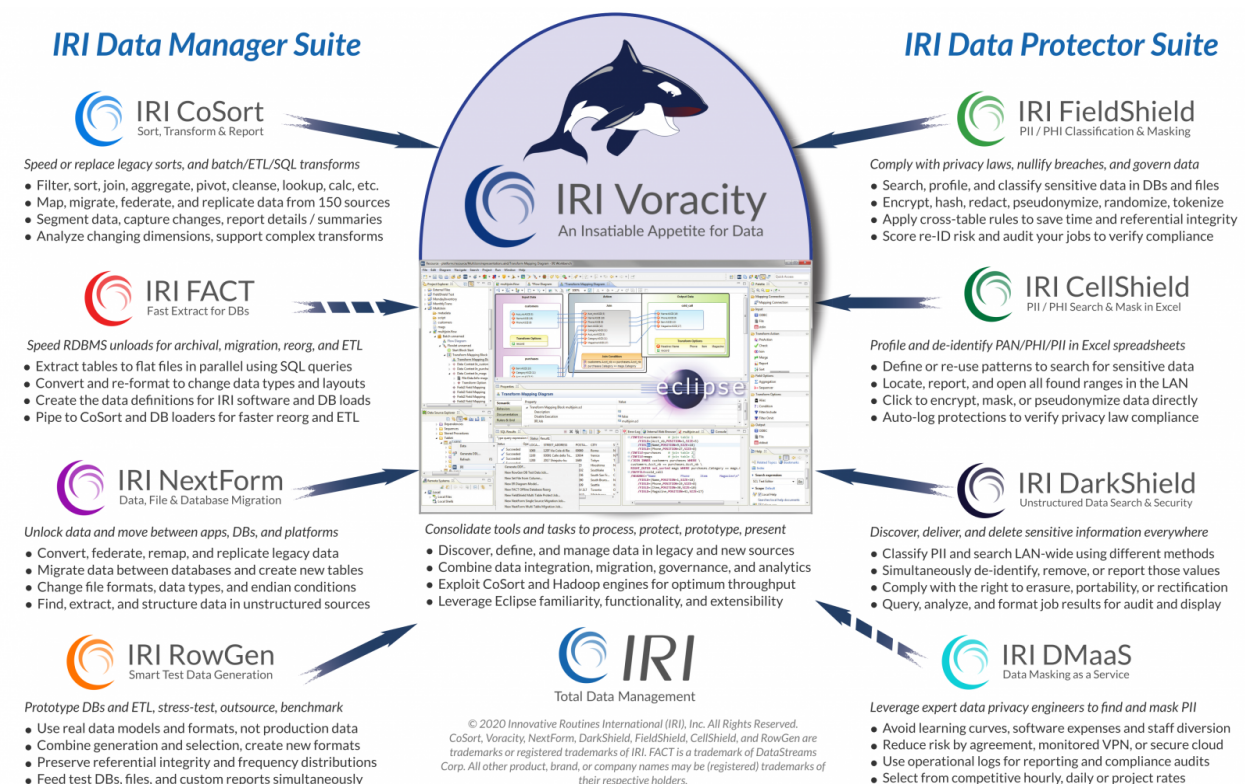
Introduction	1
Solution Architecture	2
TDH User Roles	3
Project Set Up	3
Configuring Test Data Sets (Datasets)	4
Configuring Datasets Using IRI Voracity Job Types	5
Configuring Datasets Using a Database	6
Database Masking	6
Configuring Datasets Using an API Source	7
Configuring Datasets Using Synthetic Data	7
Configuring Datasets Using Workflows	8
Configuring Datasets Using a Web Bot	8
Data & Database Subsetting	9
Job Scheduling	10
Lock & Unlock	10
Notification Service	11
Coach Marks	11
TDH Assistant	12
Interface Customization	12

Introduction

TDH – The [ValueLabs Test Data Hub \(TDH\)](#) is a web-based platform which provides test data creation and management solutions for modern development and testing teams. The objective of TDH is to automate test data creation so testers have readily available production-quality test data in desired frequencies and volumes. TDH also supports IRI Voracity test data creation (data masking, subsetting and synthesis).



IRI – [IRI Voracity](#) platform targets are now among the data sources built into TDH to provision test data. Voracity-included [IRI FieldShield](#) (or [DarkShield](#)) [data masking](#), [DB subsetting](#), or [IRI RowGen](#) smart [data synthesis](#) jobs are designed and run in the [IRI Workbench](#) (Eclipse) IDE, or in some cases in TDH directly.



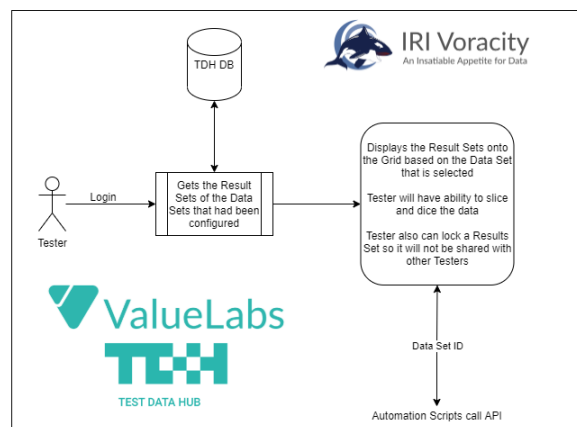
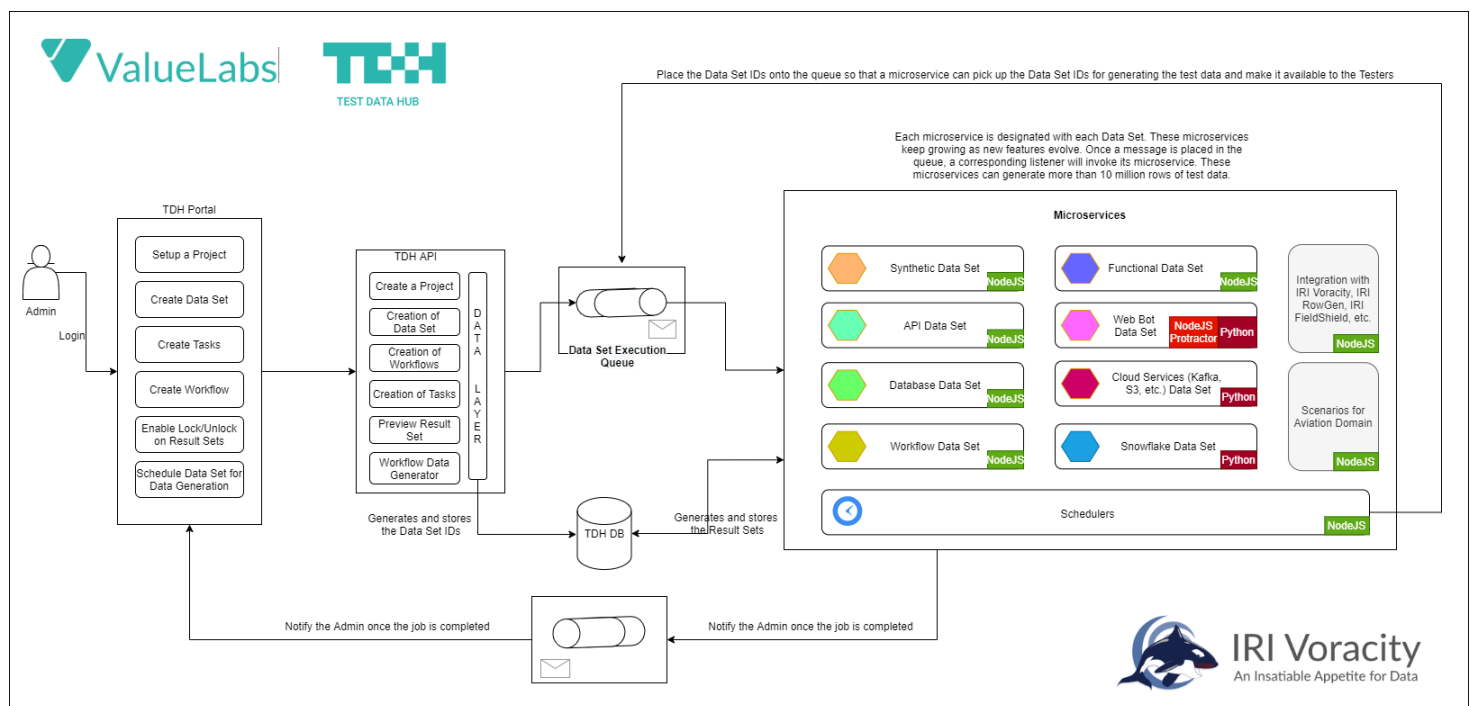
ValueLabs and IRI have come together to provide a comprehensive and affordable Test Data Generation and Management *solution* that supports existing and new test data management strategies through a simple and clean web-based solution with an intuitive UI/self-service portal for on-demand test data.

Solution Architecture

TDH provides a SSO cloud portal for administrators (managers) to configure test sets, and for testers to request and access that data on-demand. TDH uses a micro services- based architecture to speed job setup and performance, simplify deployment and troubleshooting, and update independent services.

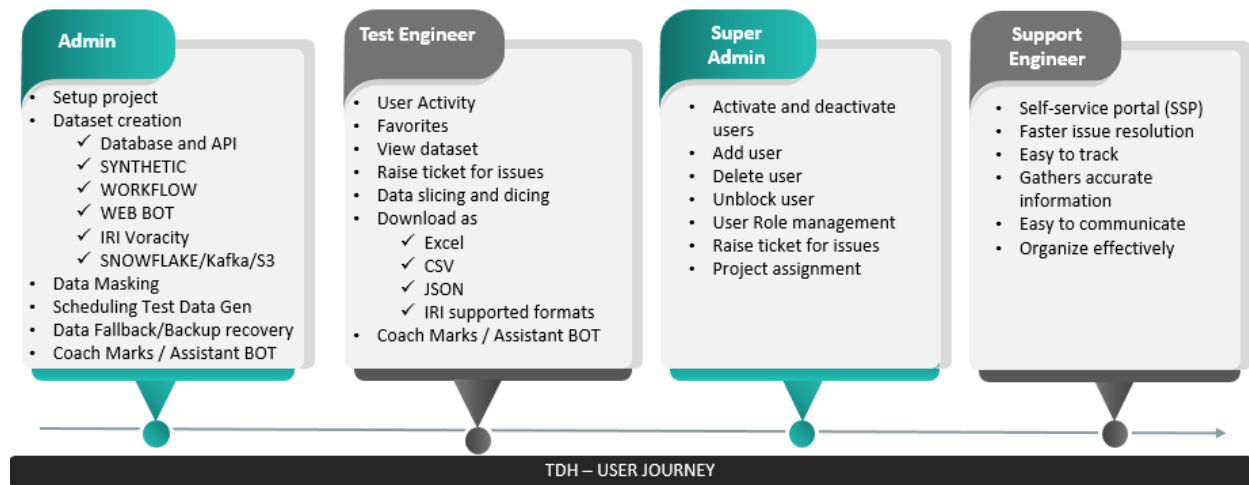
IRI Voracity is a data management platform that runs on-premise or in the cloud through an executable that parses component-product-specific job scripts to mask, subset, and synthesize (plus integrate, migrate, cleanse and/or report) on structured data. A free Eclipse front-end IDE called IRI Workbench supports graphical job design, debugging, scheduling, and sharing. Multiple APIs are also available.

TDH has built a micro service integration with IRI Voracity data masking (IRI FieldShield), DB subsetting, and test data synthesis (IRI RowGen) operations. The IRI services which produce these different forms of test data results are in-turn seamlessly configurable and managed as user-required TDH services.



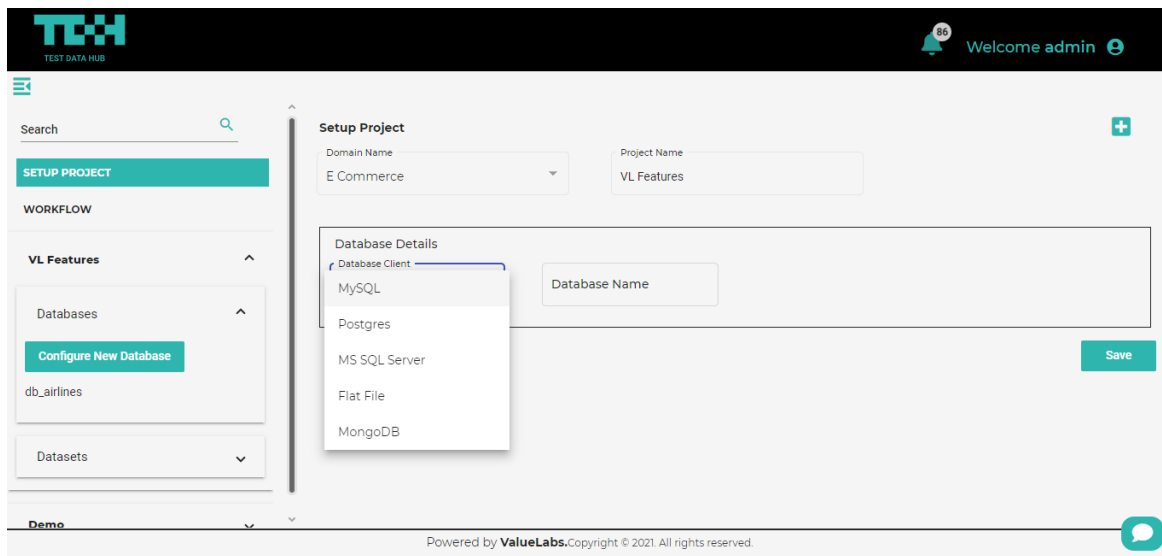
TDH User Roles

TDH user management allows administrators to manage user access to various platform features. In the user management portal, an admin can affect controls like create, view, edit, delete, and enable or disable user accounts. The below diagram shows basic roles within TDH and a snapshot of user journeys:

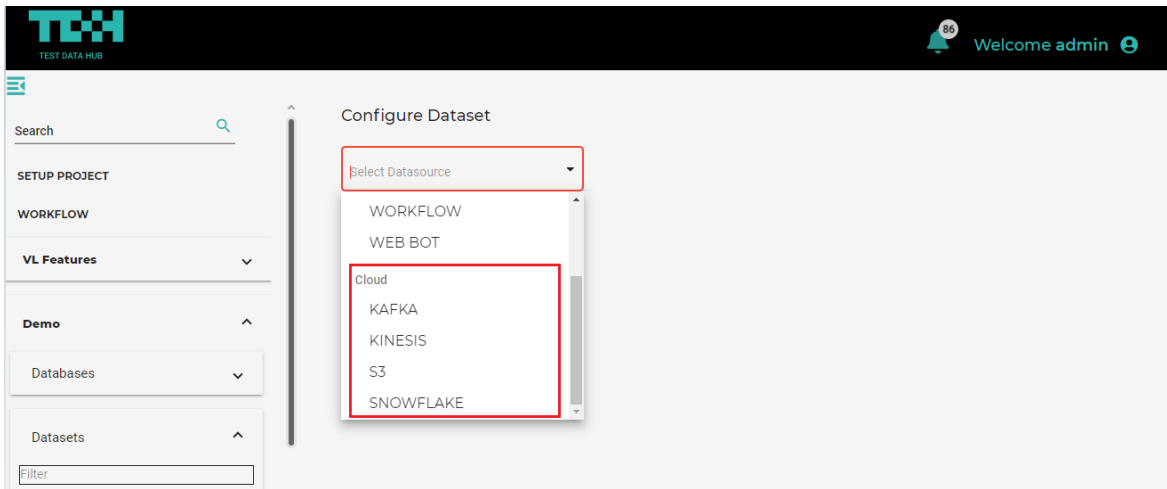


Project Set Up

TDH is architected to be highly configurable; through an intuitive UI one can quickly set up projects and configure necessary test data sets. Several database drivers are also pre-configured in TDH to save time:



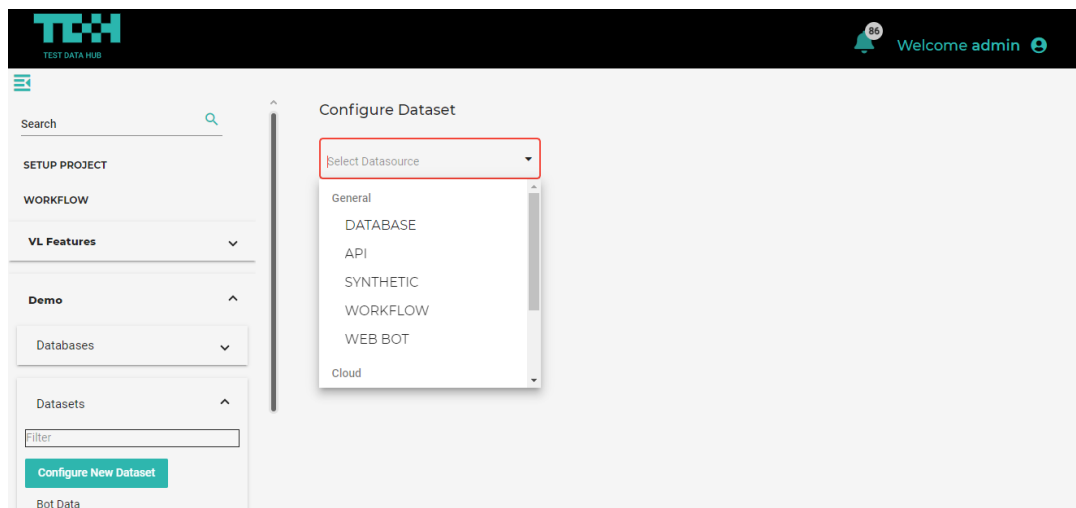
TDH also connects with different cloud services and enables scalability and business continuity. The following cloud services are pre-configured in TDH. IRI jobs also sources and target data in cloud databases, S3 buckets, Kafka and MQTT queues, pipes, custom procedures, and many file formats.



Note: Both TDH and IRI are architected to be highly configurable, so that other databases and cloud services can also be configured through custom configuration or development.

Configuring Test Sets (Datasets)

TDH users can configure datasets from multiple sources to provide test data that meets their requirements. Test sets can be sourced from: a *database* (with or without IRI FieldShield masking), *API* call, *synthetic* files or tables (via IRI RowGen jobs), a *web bot*; and, more complex workflows involving TDH and/or Voracity (i.e., multi-table DB subsetting, FieldShield, or RowGen batch scripts) job steps.



Configuring Datasets Using IRI Voracity Job Types

Browse IRI Job - Generates test data by uploading an existing IRI Voracity RowGen '.rcf' or FieldShield '.fcl' file, which is usually created in the IRI Workbench IDE or via alternative editor or IRI API call.

IRI Job Snippet - Enables the user to copy and paste '.rcf' or '.scl' code to generate relevant test data.

The screenshot shows a dropdown menu for 'Job Type' with the following options:

- Browse IRI Job - (Upload .rcf or .scl file created/used in IRI Voracity)
- IRI Job Snippet - (Copy paste .rcf or .scl code created/used in IRI Voracity)
- IRI Batch Job - (Batch job created in IRI Voracity zip and upload to generate automatic workflow)
- IRI RowGen - (Populate volumes of structurally and referentially correct test data)
- IRI FieldShield - (Classify PII centrally, find it globally, and mask it automatically)

IRI Batch Job - IRI Voracity batch jobs can be zipped and uploaded, to generate an automatic workflow of multi-table or multi-file FieldShield data masking jobs, DB subsetting (with or without masking) jobs, or RowGen data synthesis . and population jobs.

IRI RowGen - Rules for single-source RowGen (data synthesis) jobs are configured directly in TDH here.

IRI FieldShield - Rules for single-source FieldShield (data masking) jobs are configured in TDH here.

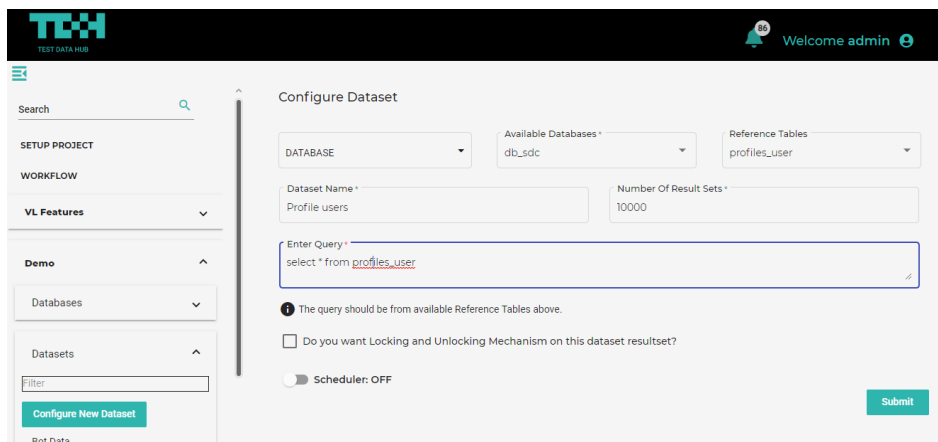
The screenshot shows the 'Configure Dataset' interface for IRI FieldShield. It includes the following fields and sections:

- Dataset Name ***: user details
- Job Type**: IRI FieldShield - Classify PII centrally, find (and report on it), and mask it automatically and consistently (deterministically)
- 1 InFile** section:
 - Process ***: Random
 - Input Name ***: user_data
 - Number Of Result Sets ***: 100000
- 2 OutFile** section (empty)
- Field Rules Table**:

Field Name	Type	Length	Separator	Set	Rules	Actions
Creditcard	DIGIT	20	Space	None	Credit Card	
CVV	DIGIT	3	Space	None	Hashing	+ -

Configuring Datasets Using a Database

Enables users to select a database from available databases and provides a provision to write SQL queries or stored procedures to SELECT specific sets of data for testing. This can of course also include data in tables that IRI processes have previously masked, subsetted, or generated.

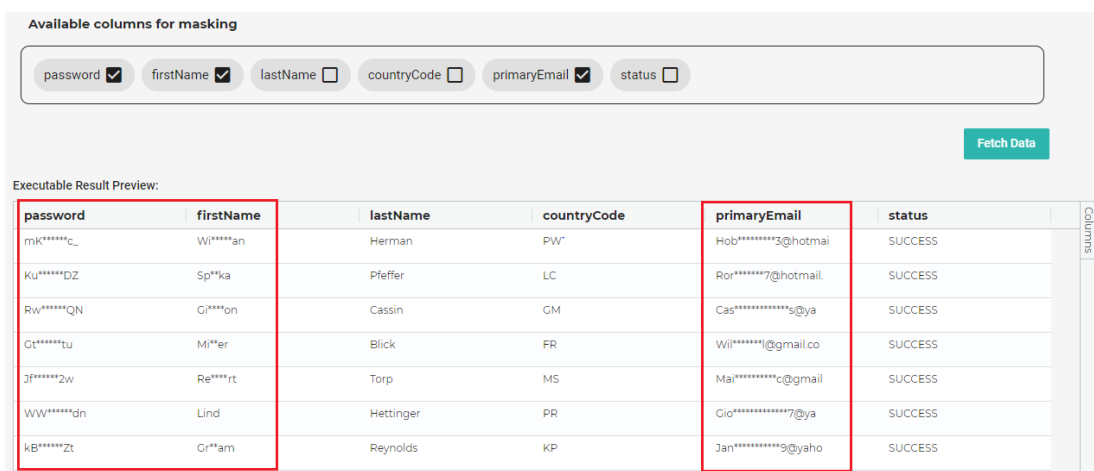


Database Masking

To de-identify or anonymize sensitive product data for test purposes, and to comply with data privacy laws, TDH-IRI users can decide which fields need to be protected and how.

[IRI FieldShield](#) users in IRI Workbench classify PII centrally, find it globally, and mask it automatically and consistently to preserve realism and referential integrity. Encryption, pseudonymization, redaction, and other functions are available and saved as rules to apply in both production and test environments. TDH users can browse, modify, and run these [Workbench-created \(typically multi-table batch\) jobs](#).

TDH also provides a simple and intuitive user interface for configuring single-source FieldShield data masking jobs on-demand, directly from the portal:



password	firstName	lastName	countryCode	primaryEmail	status
mK*****c_	W*****an	Herman	PW'	Hob*****2@hotmail	SUCCESS
KU*****DZ	Sp**ka	Pfeffer	LC	Ror*****7@hotmail.	SUCCESS
RW*****QN	Gi***on	Cassin	GM	Cas*****s@ya	SUCCESS
Qt*****tu	Mi**er	Blick	FR	Wil*****l@gmail.co	SUCCESS
Jf*****2w	Re****rt	Torp	MS	Mai*****c@gmail	SUCCESS
WW*****dn	Lind	Hettinger	PR	Gio*****7@ya	SUCCESS
kB*****Zt	Gr**am	Reynolds	KP	Jan*****9@yaho	SUCCESS

Configuring Datasets Using an API Source

TDH users can also build test data sets from their own, or third-party, applications through an API call. To use this feature, they select the required API type from the drop-down menu, and enter the relevant details to establish the connection and fetch the relevant test data from the API.

These API source controls allow users to submit a refresh of a dataset (PUT), update the status of a currently refreshing dataset (POST), view the history of a dataset, or DELETE the dataset. TDH also supports other (own/third-party) API scenarios via an API end-point and its details.

Finally, the API dataset can also be used for API testing to validate the functionality, reliability, performance, and security of the programming interfaces (API) themselves.

The screenshot shows the 'Configure Dataset' form in the TDH interface. The 'Type Of API' dropdown menu is open, showing options: GET, POST, PUT, and DELETE. The form includes fields for 'Dataset Name' (Users), 'URL' (http://jsonplaceholder.typicode.com/users), and 'Filter String'. There are checkboxes for 'Do you want Locking and Unlocking Mechanism on this dataset resultset?' and a 'Scheduler' toggle set to 'OFF'. A 'Submit' button is at the bottom right.

Configuring Datasets Using Synthetic Data

Test data is artificially created rather than being generated by actual events. TDH uses IRI RowGen to create rich synthetic data from scratch, reducing the need to use production data in testing.

The screenshot shows the 'Configure Dataset' form in the TDH interface for synthetic data. The 'Type' dropdown is set to 'SYNTHETIC'. The 'Dataset Name' is 'User-Emails'. The 'Number Of Result Sets' is 100000. The 'Type of Process' is 'Random'. There are three field configurations: 'First Name' (Type: ALPHA, Length: 40, Set: firstNa...), 'Last Name' (Type: ALPHA, Length: 40, Set: lastNa...), and 'E Mail' (Type: ALPHA_D..., Length: 30, Set: email). There are checkboxes for 'Do you want Locking and Unlocking Mechanism on this dataset resultset?' and a 'Submit' button.

Configuring Datasets Using Workflows

This feature mimics user business processes and logic in configuring custom workflows that generate relevant test data. Using the task builder can save roughly 40% of the effort involved in every sprint.

For example, to stress-test user login functionality for a website, a QA or software engineer can use the Workflow feature in TDH to eliminate the manual work of creating user login details every time. That tester can reflect the necessary business logic to accomplish that in four TDH Workflow tasks like these:

Task 1 - Create bulk users using the default synthetic task option;

Task 2 - Generate a token against each user to validate user details by using API task option;

Task 3 - Send the user details along with token to the Register API , which handles validation of the user details and token; and,

Task 4 - Once the user details are passed into the Register API (Task 3), they are saved in a database where the user registration can be validated through a written functional task option.

The screenshot displays the 'Configure Dataset' page in the Test Data Hub (TDH) application. The interface is divided into a left sidebar and a main content area. The sidebar contains navigation links: 'SETUP PROJECT', 'WORKFLOW', 'VL Features', 'Demo', 'Databases', and 'Datasets'. The 'Datasets' section is expanded, showing a 'Filter' input and a 'Configure New Dataset' button. The main content area is titled 'Configure Dataset' and features a 'Workflow' dropdown menu set to 'User login info'. Below this, the 'Sequence of Steps' section shows a vertical flow of four tasks: 'Customer reg details', 'Token Generate', 'Status of Registration', and 'Save user login info', connected by downward arrows. At the bottom of the main area, there are input fields for 'Dataset Name' (containing 'Login-Data') and 'Number Of Sets' (containing '10').

Configuring Datasets Using a Web Bot

TDH also provides an AI-enabled bot that can scan any website and retrieve necessary field information. This accelerates the test data generation process by removing manual intervention in creating test data.

For example, an e-commerce application developer may need test data with relevant attributes that already exist on other e-commerce web sites, say for a project involving sales data analytics to help the company understand competing product offerings. The built-in web bot feature in TDH can be configured to automatically extract (scrape) feature and price attributes from product information on other sites and populate new test sets with that data. Testers no longer need scour and rebuild such data by hand.

Search

Dataset Name: Web Bot test

SETUP PROJECT

WORKFLOW

VL Features

Databases

Datasets

web

Configure New Dataset

test-web

web bot 1

WebbtSHF

Web Bot test

WEB BOT

Dataset Name

Web Bot test

features/login.feature x

GIFTS x

NEW ARRIVALS x

Best Dad x

Number Of Result Sets

1

Do you want Locking and Unlocking Mechanism on this dataset resultset?

Scheduler: OFF

Fetch Data

Data & Database Subsetting

Another way the TDH-IRI solution provides users faster, on-demand access to test data is through valid subsets of relational database tables or flat files. The TDH process shown below is one way to do this:

Search

SETUP PROJECT

WORKFLOW

Demo

Databases

Configure New Database

Datasets

Setup Project

Domain Name

E Commerce

Project Name

Demo

Database Details

Database Client

Postgres

Database Name

db_sdc

DB Host Address

10.10.53.130

DB User Name

DB Password

DB Port

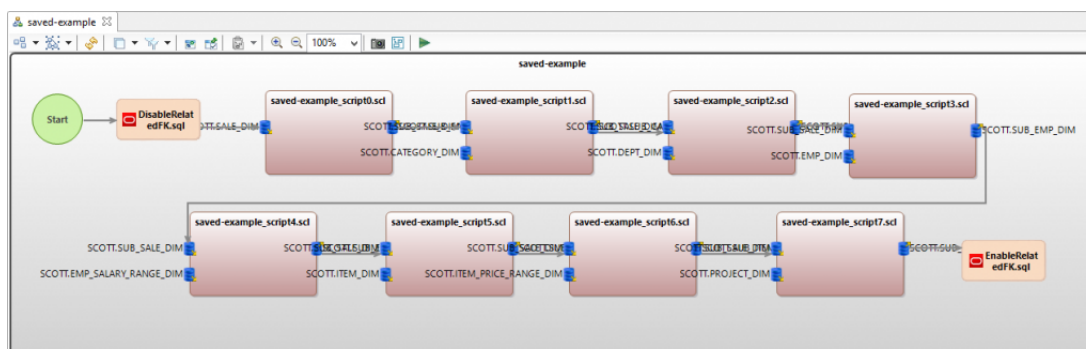
5432

Fetch Tables

Available tables for the requested database. Please select tables to import into TDH database

Table Name	Select	Edit
api_users	<input checked="" type="checkbox"/>	
cases_case	<input checked="" type="checkbox"/>	
cases_patient	<input checked="" type="checkbox"/>	
commerce_order	<input type="checkbox"/>	

Another way is via the IRI process described [here](#), and shown in the IRI Workbench diagram below,



which TDH can also invoke as a batch process once built. Either method simply and efficiently extracts small, referentially correct test sets, removing the dependencies on DBAs and experts to provision them.

Job Scheduling

TDH automation functionality works 24/7 behind the scenes to ensure that test data is generated in desired volumes and frequencies. It also helps testing teams plan and manage current and future test data generation requirements.

The screenshot displays the 'WORKFLOW' section of the TDH interface. On the left, a sidebar contains 'VL Features', 'Databases', and 'Datasets' sections. The main area is titled 'SYNTHETIC' and includes a 'syn save edit' button. Below this, there are fields for 'Number Of Result Sets' (set to 3) and 'Type of Process' (set to Random). A table lists three fields: 'fname', 'lname', and 'phone', each with a 'Type' of ALPHA and a 'Length' of 5. To the right of each field is a 'Set' dropdown (firstNa..., lastNa..., phone...) and a checkbox. At the bottom, a green-bordered box highlights the scheduling options. It includes a 'Scheduler: ON' toggle, two radio buttons for 'Overwrite data with the existing result set' (selected) and 'Append data to an existing result set', and two rows of 'Frequency' (Daily) and 'Time' (14:47 and 23:47) settings, each with a clock icon and add/remove buttons.

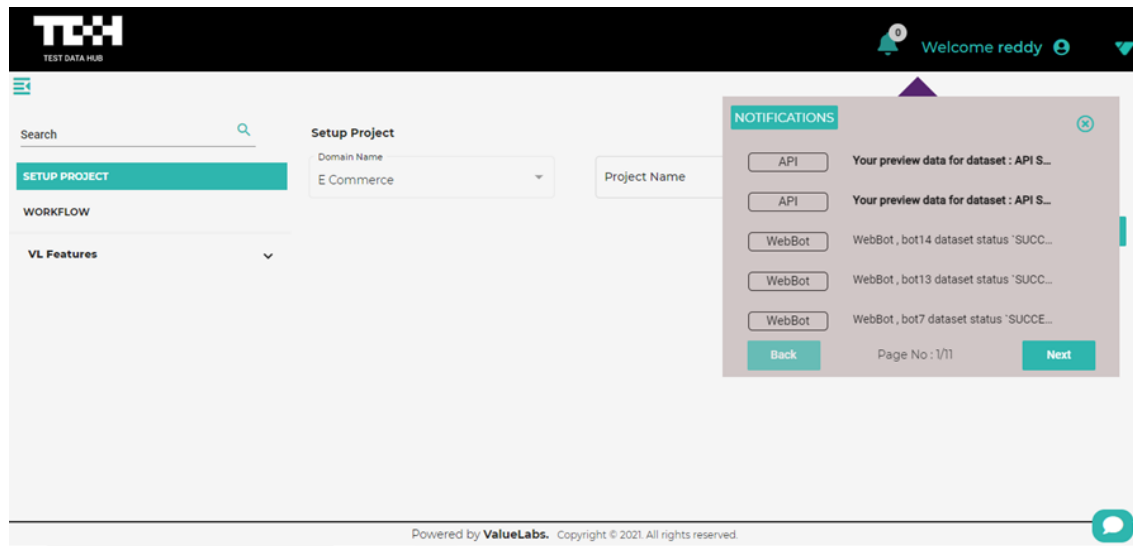
Lock & Unlock

TDH gives every tester the ability to lock unique records, preventing others from using the same dataset:

This screenshot shows the same TDH interface as the previous one, but with a green-bordered box highlighting a checkbox labeled 'Do you want Locking and Unlocking Mechanism on this dataset resultset?'. The checkbox is checked. To the right of this box, the text 'Lock, Un Lock' is visible. The rest of the interface, including the sidebar and the synthetic data configuration fields, remains the same.

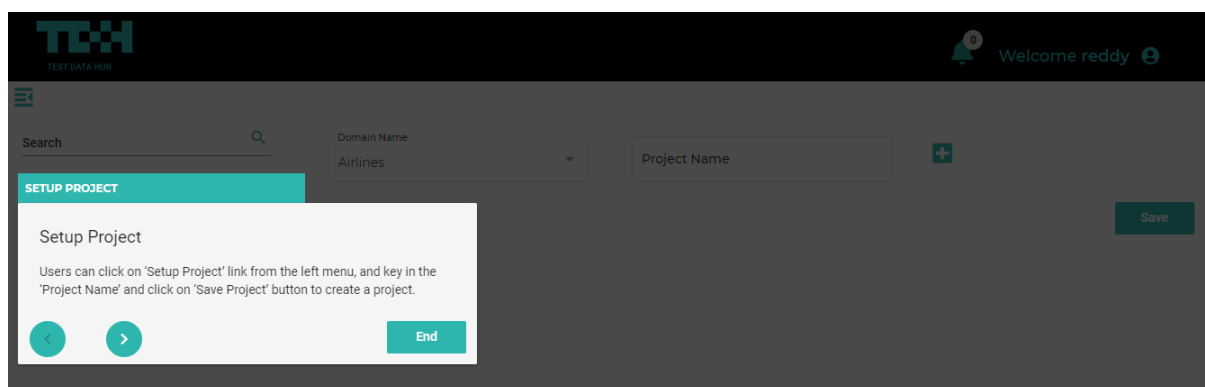
Notification Service

TDH has a strong notification engine to help users stay on top of their test data generation activities, and to improve communication between TDH users. This service helps users define and track their progress, and prompts them in timely fashion so that they can better manage their work:



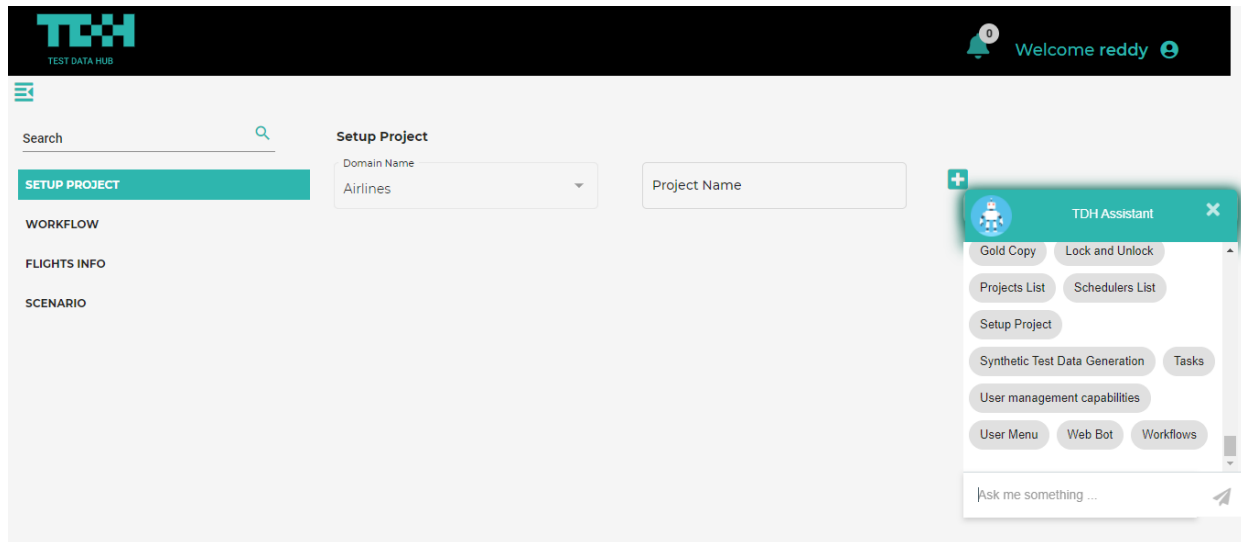
Coach Marks

TDH coach marks are context-sensitive pop-up tips to help users learn and work with TDH features. They save time and training costs, address multiple change management issues, and improve user adoption. This feature is useful for both existing and new capability explanations.



TDH Assistant

Similarly, the TDH Assistant feature enables users to become self-sufficient by educating them, and getting them up to speed on using TDH functionality. This simultaneously increases user productivity and reduces the dependency on manuals or trainers.



Interface Customization

TDH adapts to any client branding easily. CSS preprocessors can apply new brand guidelines in one go.

