

**API Developer Manual
Version 2.0.4**

Krux

Drill more, faster, smarter



Contents

Introduction	1
API Usage flow	1
Service Account Creation	2
Authorization	2
Token Expiration	2
Authentication Endpoint	3
Plan API Suite	4
Hole and Plan/Project Relationship Diagram.....	4
LOOKUP/GET Methods	4
PlanTypes GET	4
Locations GET	5
Currencies GET	6
Plans GET	7
Projects GET	8
HoleTypes GET	9
Tenement GET	10
SecondaryStatus GET	11
Holes GET	11
POST Methods	16
Plan POST	16
Errors and Status codes	16
Holes POST	17
Errors and Status codes	19
Field Validations.....	19
PATCH Methods.....	20
Plan PATCH.....	20
Errors and Status codes	21
Field Validations.....	21
Holes PATCH.....	22
Errors and Status codes	24
Field Validations.....	24



DELETE Methods	26
Plan DELETE	26
Holes DELETE	27
Export API Suite	28
Companies GET	28
ExportTypes GET	29
Retrieve Incremental Hole Data	32
GetData GET	32
Working with the API Data	34
UIDs	34
Processing Incremental Data	34
Using Report and Charge Data	35
Hole Data	37
Hole UID Format	37
Daily Shift Report (DSR) Data	38
DSR Data Definitions	38
DSR UID Format	38
DSRWorkerLabour UID Format	39
DSRActivity UID Format	40
DSRActivityEquipment UID Format	42
DSRActivityLabour UID Format	42
DSREquipmentUsage UID Format	43
DSRConsumables UID Format	44
DSRFluids UID Format	44
DSRSamples UID Format	45
DSRPerDrillRate UID Format	45
DSRAdditionalCharges UID Format	46
DSRActivityCodes UID Format	46
DSRCoreRun UID Format	47
DSRDirectionalRun UID Format	48
Timesheet (TS) Data	50
TS UID Format	50
TSWorkersLabour UID Format	51

TSActivityLabour UID Format	52
TSActivityEquipment UID Format	52
TSConsumables UID Format.....	53
TSFluid UID Format	54
Equipment Rentals Data	55
EquipmentRental UID Format	55
Additional Charges Data.....	55
AdditionalCharges UID Format.....	55
Appendix	57
Status Codes.....	57
Error Codes	57

Introduction

This guide is designed to help subscribed users understand and utilize the powerful capabilities of the Krux platform through its API offerings. The Krux API Suite provides integration options for importing and exporting data, enabling users to efficiently manage and exchange operational data with other systems.

The suite is divided into two main components:

1. **[Plan API](#)**: Allows users to import plans and hole data directly into the Krux platform, ensuring that all planning and drilling information is up-to-date and centrally accessible.
2. **[Export API](#)**: Enables users to retrieve incremental report data from Krux, making it easy to synchronize Krux data with external systems or databases.

Whether you're integrating Krux with third-party applications or automating internal workflows, this documentation will guide you through the process with detailed endpoints, request/response structures, and usage examples.

API Usage flow

The following steps constitute the full flow of the API and gives you a general idea on how the APIs can be best used efficiently

User Authentication Steps

1. Client sends a POST request to the Krux Authentication Endpoint with service account user credentials (e.g., username/password or API key).
2. Krux returns an Access Token (Bearer Token).
3. Client includes the Access Token in the Authorization header for all subsequent API calls.

Plan API Suite Usage (Import Plans and Holes into Krux)

1. Client prepares plan data in the required JSON format.
2. Client sends a GET request to `/api/V1/lookups` to get appropriate lookup IDs needed for a valid import into Krux
3. Client sends a POST request to `/api/v1/Plans` endpoint with plan data in the request body.
4. Client prepares hole data in the required JSON format.
5. Client sends a GET request to `/api/V1/lookups` to get appropriate lookup IDs needed for a valid import into Krux
6. Client sends a POST request to `/api/v1/Holes` endpoint with hole data in the request body.
7. Krux validates the data and stores it in the system against a plan or project or both.
8. API returns a success response or an error message with details if validation fails.

Export API Suite Usage (Export Data out of Krux)

1. Client sends a GET request to `/api/export/data` with appropriate filters (e.g., ExportDateTime, query types).
2. Krux processes the request and retrieves incremental data.
3. API returns data payload in JSON format.



4. Client system parses and ingests the data for downstream use.

Service Account Creation

A Service Account is required to use the API and can be created using the following steps.

- Login to KruxMetrix using account with Administrator privileges.
- Navigate to Settings -> Users.
- Press Add User button which will open the New User page.
- On the New User page, check the Service Account checkbox and enter the required fields. Click Save to continue.

☒ Service Account

Service Account Type

SSO Service Account

Account Name

Save

Cancel

Authorization

This API utilizes Bearer Tokens for authorization. The Bearer Token must be passed in a key named "Authorization" in each subsequent call to the API.

AUTHORIZATION

Bearer Token

Token

<token>

Token Expiration

Tokens are valid for 7 days. Requests made using an expired token will result in 401 Unauthorized responses from the server and the Response Headers will contain the **WWW-Authenticate** key that has a value similar to **Bearer error="invalid_token", error_description="The token expired at '01/01/2025 17:57:27'"**



Authentication Endpoint

Used to authenticate user and obtain the Bearer Token from “token” in the Json response.
The token key returned in a successful authentication response is needed for all subsequent API calls.

ENDPOINT

<https://metrixapi.kruxanalytics.com/users/auth>

BODY

```
{"Username": "service_account", "Password": "service_account_password"}
```

RESPONSE

```
{
  "$id": "1",
  "id": 12345,
  "firstName": "Service",
  "lastName": "Account",
  "username": "service_account",
  "password": null,
  "token": "your_token",
  "locked": false,
  "companyID": 12345,
  "configurationName": null,
  "configurationID": 0,
  "subscriptionList": "KruX Api",
  "roleList": "API Service Account",
  "companyTypeId": 1
}
```

Errors

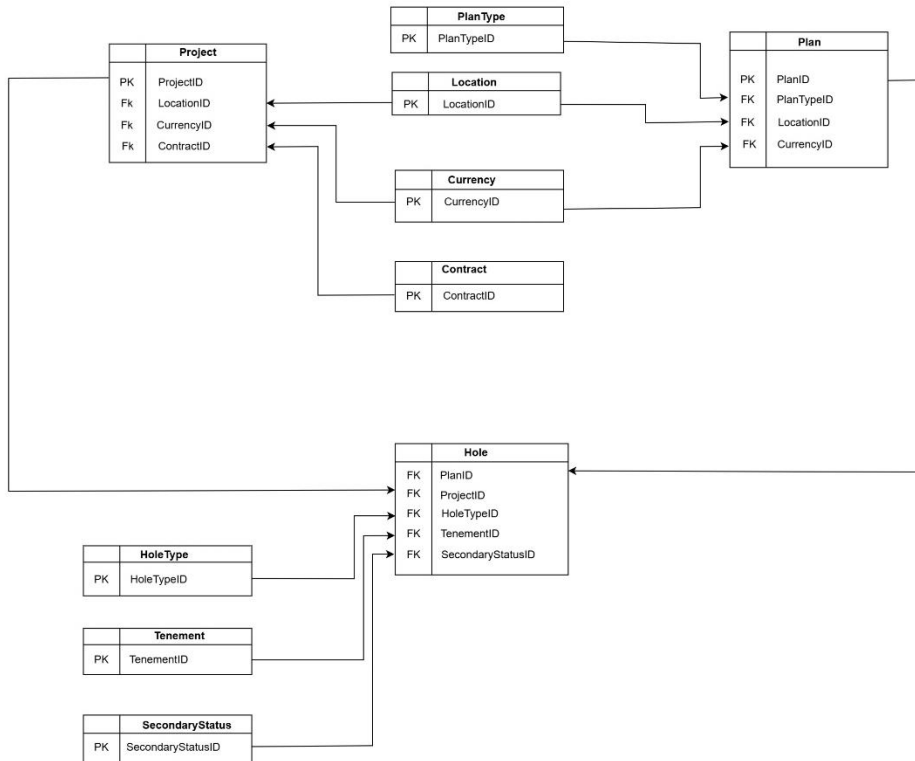
Errors occurring while authenticating will result in 401 Unauthorized.

```
{
  "$id": "1",
  "message": "Username or password is incorrect"
}
```

Plan API Suite

Hole and Plan/Project Relationship Diagram

This relationship diagram details unique IDs needed to create Plans or Holes in the Krux system:



LOOKUP/GET Methods

These methods are only available to Resource Companies. It will return unauthorized error for Drilling company accounts.

PlanTypes **GET**

This endpoint can be used to request non-deleted PlanType data available to the company of the authorized user.

ENDPOINT

<https://metrixapi.kruxanalytics.com/api/v1/Plans/types>

AUTHORIZATION

This request uses the Bearer Token

HEADERS

None



REQUEST PARAMETERS

None

Response

```
[
  {
    "PlanTypeID": 21,
    "PlanType": "Plan-June",
    "Status": "ACTIVE"
  }
]
```

Data Definition

	PlanTypeID	INT
	PlanType	NVARCHAR(100)
	Status	VARCHAR(30)

Locations **GET**

This endpoint can be used to request non-deleted Location data available to the company of the authorized user.

ENDPOINT

<https://metrixapi.kruxanalytics.com/api/v1/Lookups/locations>

AUTHORIZATION

This request uses the Bearer Token

HEADERS

None




REQUEST PARAMETERS

None

Response

```
[
  {
    "locationID": 0,
    "locationName": "string",
    "abbreviation": "string",
    "status": "string"
  }
]
```

Data Definition

 LocationID	INT
LocationName	NVARCHAR(100)
Abbreviation	VARCHAR(10)
Status	VARCHAR(30)

Currencies **GET**

This endpoint can be used to request available Currencies in the Krux system for an authorized user.

ENDPOINT

<https://metrixapi.kruxanalytics.com/api/v1/Lookups/currencies>

AUTHORIZATION

This request uses the Bearer Token

HEADERS

None

REQUEST PARAMETERS

None

Response

```
[
  {
    "currencyID": 0,
    "country": "string",
  }
]
```



```

        "currency": "string",
        "currencyCode": "string",
        "symbol": "string"
      }
    ]
  }
}

```

Data Definition

	CurrencyID	INT
	Country	VARCHAR(100)
	Currency	NVARCHAR(100)
	CurrencyCode	VARCHAR(100)
	Symbol	NVARCHAR(100)

Plans **GET**

This endpoint can be used to request non-deleted Plans available to the company for the authorized user. If PlanID is provided, then details about that plan is retrieved. If no parameter is provided, all the plans are retrieved.

ENDPOINT

<https://metrixapi.kruxanalytics.com/api/v1/Plans>

AUTHORIZATION

This request uses the Bearer Token

HEADERS

None

REQUEST PARAMETERS

{id}

Response


```

[
  {
    "planId": 0,
    "planName": "string",
    "status": "string",
    "locationID": 0,
    "locationName": "string",
    "createDateTime": "2025-04-02T22:37:27.065Z"
  }
]

```



Data Definition

	MineProjectID	INT
	MineProjectName	NVARCHAR(200)
	Status	VARCHAR(30)
	LocationID	INT
	LocationName	VARCHAR(100)
	CreateDateTime	DATETIME

Projects **GET**

This endpoint can be used to request the list of non-deleted Projects available to the company of the authorized user.

ENDPOINT

<https://metrixapi.kruxanalytics.com/api/v1/Lookups/projects>

AUTHORIZATION

This request uses the Bearer Token

HEADERS

None

REQUEST PARAMETERS


None

Response

```
[
  {
    "projectID": 0,
    "contractID": 0,
    "createDateTime": "2025-04-02T22:47:51.006Z",
    "projectName": "string",
    "contractName": "string",
    "poNumber": "string",
    "locationID": 0,
    "locationName": "string",
    "projectStatus": "string"
  }
]
```

Data Definition



 ProjectID	INT
ProjectName	NVARCHAR(100)
Status	BIT
ContractID	INT
ContractName	NVARCHAR(100)
PO Number	NVARCHAR(255)
LocationID	INT
LocationName	VARCHAR(100)
CreateDateTime	DATETIME

HoleTypes **GET**

This endpoint can be used to request the list of non-deleted Hole Types available to the company of the authorized user.

ENDPOINT

<https://metrixapi.kruxanalytics.com/api/v1/Holes/types>

AUTHORIZATION

This request uses the Bearer Token

HEADERS

None

REQUEST PARAMETERS

None

Response

```
[
  {
    "holeTypeID": 0,
    "holeType": "string",
    "status": "string"
  }
]
```

Data Definition



	HoleTypeID	INT
	HoleType	NVARCHAR(200)
	Status	VARCHAR(30)

Tenement GET

This endpoint can be used to request the list of non-deleted Tenements available to the company of the authorized user.

ENDPOINT

<https://metrixapi.kruxanalytics.com/api/v1/Lookups/tenements>

AUTHORIZATION

This request uses the Bearer Token

HEADERS

None

REQUEST PARAMETERS

None

Response

```
[
  {
    "tenementID": 0,
    "tenementName": "string",
    "status": "string",
    "startDate": "2025-04-02",
    "endDate": "2025-04-02"
  }
]
```

Data Definition



	TenementID	BIGINT
	TenementName	NVARCHAR(255)
	StartDate	DATE
	EndDate	DATE
	Status	NVARCHAR(10)

SecondaryStatus **GET**

This endpoint can be used to request the list of non-deleted SecondaryStatuses available to the company of the authorized user.

ENDPOINT

<https://metrixapi.kruxanalytics.com/api/v1/Lookups/secondaryStatuses>

AUTHORIZATION

This request uses the Bearer Token

HEADERS

None


REQUEST PARAMETERS

None

Response

```
[
  {
    "secondaryStatusID": 0,
    "secondaryStatus": "string",
    "status": "string"
  }
]
```

Data Definition

	SecondaryStatusID	INT
	SecondaryStatus	NVARCHAR(100)
	Status	VARCHAR(30)

Holes **GET**

This endpoint can be used to request the list of Holes and associated data available to the company of the authorized user.



- If no PlanID or ProjectID is provided as parameter, return all holes for the company.
- If the ProjectID or PlanID is passed, retrieve data for all holes under that Plan or Project.
- The id parameter can be either a PlanID or ProjectID based on the isPlan boolean.

ENDPOINT

<https://metrixapi.kruxanalytics.com/api/v1/Holes?isPlan=true>

AUTHORIZATION

This request uses the Bearer Token

HEADERS

None

REQUEST PARAMETERS


Key	Value	Description
id	Integer (12345)	The id is the ProjectID or PlanID filter for the data that is going to be returned. If no value is provided here, all holes for the company is retrieved
isPlan	boolean (true, false)	If isPlan is true, the id value is parsed as a PlanID and if isPlan is false, the id is parsed as a ProjectID

Response

```
[
  {
    "holeID": 14-0,
    "holeName": "string",
    "holeStatus": "string",
    "completeDateTime": "2025-04-02T23:17:56.594Z",
    "holeType": "string",
    "contractorCompany": "string",
    "contracteeCompany": "string",
    "contract": "string",
    "project": "string",
    "plan": "string",
    "firstActivityDate": "2025-04-02",
    "lastActivityDate": "2025-04-02",
    "maxDepth": 0,
    "plannedDepth": 0,
    "depthUnit": "string",
    "totalDistanceDrilled": 0,
    "totalActivityHours": 0,
    "totalDrillingHours": 0,
    "penetration": 0,
    "easting": 0,
    "northing": 0,
    "utmZone": "string",
    "mineGridEasting": 0,
```

```
"mineGridNorthing": 0,  
"elevation": 0,  
"elevationUnit": "string",  
"plannedAzimuth": 0,  
"plannedDip": 0,  
"deletedFlag": "string",  
"contractID": 0,  
"firstDrillingActivityDate": "2025-04-02",  
"planID": 0,  
"projectID": 0,  
"plannedHoleName": "string",  
"holeSpecifications": "string",  
"tenement": "string",  
"isDirectional": true,  
"targetNorthing": 0,  
"targetEasting": 0,  
"targetElevation": 0,  
"directionalUnit": "string",  
"plannedDogleg": 0,  
"plannedGTF": 0,  
"startType": "string",  
"directionalDepthFrom": 0,  
"directionalDepthTo": 0,  
"directionalInclinationStart": 0,  
"directionalInclinationEnd": 0,  
"directionalAzimuthStart": 0,  
"directionalAzimuthEnd": 0,  
"SecondaryStatus": 0,  
"drillPriority": 0  
}  
]
```

Data Definition

 UID	VARCHAR(22)
HoleID	INT
HoleName	NVARCHAR(100)
HoleStatus	VARCHAR(10)
CompleteDateTime	DATE
HoleType	NVARCHAR(200)
ContractorCompany	VARCHAR(200)
ContracteeCompany	VARCHAR(200)
Contract	NVARCHAR(100)
Project	NVARCHAR(100)
Plan	NVARCHAR(200)
FirstActivityDate	DATE
LastActivityDate	DATE
MaxDepth	DECIMAL(22, 2)
PlannedDepth	DECIMAL(22, 2)
DepthUnit	NVARCHAR(100)
TotalDistanceDrilled	DECIMAL(22, 3)
TotalActivityHours	DECIMAL(22, 3)
TotalDrillingHours	DECIMAL(22, 3)
Penetration	DECIMAL(22, 3)
Easting	FLOAT
Northing	FLOAT
UTMZone	NVARCHAR(50)
MineGridEasting	FLOAT
MineGridNorthing	FLOAT
Elevation	FLOAT
ElevationUnit	NVARCHAR(100)
PlannedAzimuth	FLOAT
PlannedDip	FLOAT
DeletedFlag	VARCHAR(1)
ExportDateTime	DATETIME
ContractID	INT
FirstDrillingActivityDate	DATE
PlanID	INT
ProjectID	INT
PlannedHoleName	NVARCHAR(100)
HoleSpecifications	NVARCHAR(4000)
Tenement	BIGINT
IsDirectional	BIGINT
TargetNorthing	FLOAT
TargetEasting	FLOAT
TargetElevation	FLOAT
DirectionalUnit	INT
PlannedDogleg	FLOAT
PlannedGTF	FLOAT
StartType	INT
DirectionalDepthFrom	FLOAT
DirectionalDepthTo	FLOAT
DirectionalInclinationStart	FLOAT
DirectionalInclinationEnd	FLOAT
DirectionalAzimuthStart	FLOAT
DirectionalAzimuthEnd	FLOAT
DrillPriority	INT



POST Methods

These methods are only available to Resource Companies. These endpoints will allow submitting Plans and Holes into Krux.

Plan POST

This method can be used to insert plans into Krux

ENDPOINT

<https://metrixapi.kruxanalytics.com/api/v1/Plans>

AUTHORIZATION

This request uses the Bearer Token

HEADERS

None

REQUEST PARAMETER

None

REQUEST BODY SAMPLE

```
{
  "plans": [
    {
      "planName": "AutoTest-12345",
      "planTypeId": 41,
      "locationId": 242,
      "budget": 1000,
      "currencyId": 1,
      "plannedDepth": 20,
      "plannedDepthUnit": "feet"
    }
  ]
}
```

Response Body

```
{"InvalidRecords":[],"Status":1}
```

Errors and Status codes

The error and status codes can be referenced in the [APPENDIX](#)



When inserting Plans into Krux, there are specific restrictions governing the values that may be entered into each field. These restrictions are intended to ensure data consistency and integrity during Hole creation. Please ensure all Plan creation entries strictly adhere to the allowed values for each field. Invalid entries will result in errors or rejected submissions.

Below is the complete list of fields along with their corresponding allowed values:

Fields	Allowed Values
planName	Alphanumeric name that is unique to the user's company
planTypeId	Valid PlanTypeID. This can be retrieved using endpoint https://metrixapi.kruxanalytics.com/api/v1/Plans/types
locationId	Valid LocationID. This can be retrieved using endpoint https://metrixapi.kruxanalytics.com/api/v1/Lookups/locations
budget	Numerical value, allows decimals
currencyId	Valid currencyID. This can be retrieved using endpoint https://metrixapi.kruxanalytics.com/api/v1/Lookups/currencies
plannedDepth	Numerical value, allows decimals
plannedDepthUnit	This accepts one of two string values: 1. "Meters" 2. "Feet"

Holes **POST**

This endpoint can be used to Insert Holes against a Plan or Project.

ENDPOINT

<https://metrixapi.kruxanalytics.com/api/v1/Holes>

Request Body Sample – Non - Directional Hole

```
{
  "holes": [
    {
      "planId": 477,
      "projectId": 1008,
      "holeName": "NewHole1234",
      "plannedHoleName": "NewHole1234",
      "isDirectional": false,
      "plannedDepth": 150,
      "utmZone": "11N",
      "utmEasting": 13,
      "utmNorthing": 2,
      "elevation": 100,
      "plannedAzimuth": 120,
      "plannedDip": -2,
      "plannedDepthUnit": "meters",
    }
  ]
}
```



```

        "elevationUnit": "meters",
        "mineGridEasting": 44,
        "mineGridNorthing": 12,
        "holeTypeId": 43,
        "holeSpecifications": "This is a hole added from the API",
        "tenementId": 15,
        "drillPriority": "High",
        "secondaryStatusID": "26"
    }
}

```

Request Body Sample – Directional Hole:

```

{
  "holes": [
    {
      "planId": 477,
      "projectId": 1008,
      "holeName": "directionalHole",
      "plannedHoleName": "NewHole1234",
      "secondaryStatusID": "26",
      "utmEasting": 12.34,
      "utmNorthing": 12.56,
      "utmZone": "1N",
      "elevation": 450,
      "elevationUnit": "feet",
      "mineGridEasting": 10.76,
      "mineGridNorthing": 3.90,
      "isDirectional": true,
      "holeTypeId": 43,
      "holeSpecifications": "This is a directional hole added from the API",
      "tenementId": 15,
      "drillPriority": "High",
      "secondaryStatusID": "26",
      "directionalUnit": "meters",
      "targetNorthing": 20,
      "targetEasting": 40,
      "targetElevation": 2,
      "plannedDogleg": 31,
      "plannedGtf": 4,
      "startType": "Hole Bottom",
      "directionalDepthFrom": 0,
      "directionalDepthTo": 35,
      "directionalInclinationStart": 0,
      "directionalInclinationEnd": 44,
    }
  ]
}

```

```

        "directionalAzimuthStart": 20,
        "directionalAzimuthEnd": 35
      }
    ]
  }
}

```

Response Body

```

{"InvalidRecords":[],"Status":1}

```

Errors and Status codes

The error and status codes can be referenced in the [APPENDIX](#)

Field Validations

When inserting Holes against a designated Plan or Project, there are specific restrictions governing the values that may be entered into each field. These restrictions are intended to ensure data consistency and integrity during Hole creation. Please ensure all Hole creation entries strictly adhere to the allowed values for each field. Invalid entries will result in errors or rejected submissions.

Below is the complete list of fields along with their corresponding allowed values:

Fields	Allowed Values
planId	Valid PlanID. This can be retrieved using endpoint https://metrixapi.kruxanalytics.com/api/v1/Plans
projectId	Valid ProjectID. This can be retrieved using endpoint https://metrixapi.kruxanalytics.com/api/v1/Lookups/projects
holeName	Enter a Valid Hole Name that is unique in the Project/Plan the hole is being uploaded to
plannedHoleName	Enter a Valid Planned Hole Name that is unique in the Project/Plan the hole is being uploaded to
isDirectional	This accepts one of two values: 1. true 2. false
plannedDepth	Numerical value, allows decimals
utmZone	Valid UTMZone values, numerical values between 1-60 inclusive appended by N or S. Eg: 1N or 60S
utmEasting	Numerical value, allows decimals
utmNorthing	Numerical value, allows decimals
elevation	Numerical value, allows decimals
plannedAzimuth	Numerical value, allows decimals
plannedDip	Numerical value, allows decimals
plannedDepthUnit	This accepts one of two string values: 1. "Meters" 2. "Feet"



elevationUnit	This accepts one of two string values: 1. "Meters" 2. "Feet"
mineGridEasting	Numerical value, allows decimals
mineGridNorthing	Numerical value, allows decimals
holeTypeid	Valid holeTypeID. This can be retrieved using endpoint https://metrixapi.kruxanalytics.com/api/v1/Holes/types
holeSpecifications	Text field
tenementId	Valid TenementID. This can be retrieved using endpoint https://metrixapi.kruxanalytics.com/api/v1/Lookups/tenements
targetNorthing	Numerical value, allows decimals
targetEasting	Numerical value, allows decimals
targetElevation	Numerical value, allows decimals
directionalUnit	This accepts one of two string values: 1. "Meters" 2. "Feet"
plannedDogleg	Numerical value, allows decimals
plannedGtf	Numerical value, no decimals allowed
startType	Allows one of the four values: 1. Hole Bottom 2. Cut Curve 3. Wedge 4. Cement Plug
directionalDepthFrom	Numerical value, allows decimals
directionalDepthTo	Numerical value, allows decimals
directionalInclinationStart	Numerical value, allows decimals
directionalInclinationEnd	Numerical value, allows decimals
directionalAzimuthStart	Numerical value, allows decimals
directionalAzimuthEnd	Numerical value, allows decimals
drillPriority	Allows one of the four string values: 1. "Critical" 2. "High" 3. "Medium" 4. "Low"
secondaryStatusID	Valid SecondaryStatusID. This can be retrieved using endpoint https://metrixapi.kruxanalytics.com/api/v1/Lookups/secondaryStatuses

PATCH Methods

These methods are only available to Resource Companies.

Plan PATCH

This method can be used to update information on Plans

ENDPOINT



metrixapi.kruxanalytics.com/api/v1/Plans

AUTHORIZATION

This request uses the Bearer Token

HEADERS

None

REQUEST PARAMETER

None

REQUEST BODY SAMPLE

```
"plans": [
  {
    planName: "AutoTest-Edit",
    planTypeId: 41,
    locationId: 242,
    budget: 1000,
    currencyId: 1,
    plannedDepth: 20,
    plannedDepthUnit: "feet"
    planId: 786,
    planStatus: "Active"
  }
]
```

Response Body

```
{"InvalidRecords":[],"Status":1}
```

Errors and Status codes

The error and status codes can be referenced in the [APPENDIX](#)

Field Validations

When updating Plans in Krux, there are specific restrictions governing the values that may be entered into each field. These restrictions are intended to ensure data consistency and integrity during Hole creation. Please ensure all Plan creation entries strictly adhere to the allowed values for each field. Invalid entries will result in errors or rejected submissions.



Below is the complete list of fields along with their corresponding allowed values:

Fields	Allowed Values
planTypeId	Valid PlanTypeID. This can be retrieved using endpoint https://metrixapi.kruxanalytics.com/api/v1/Plans/types
locationId	Valid LocationID. This can be retrieved using endpoint https://metrixapi.kruxanalytics.com/api/v1/Lookups/locations
budget	Numerical value, allows decimals
currencyId	Valid currencyID. This can be retrieved using endpoint https://metrixapi.kruxanalytics.com/api/v1/Lookups/currencies
plannedDepth	Numerical value, allows decimals
plannedDepthUnit	This accepts one of two string values: 1. "Meters" 2. "Feet"
planStatus	This accepts one of two string values: 1. "Active" 2. "Complete"
planName	Alphanumeric name that is unique to the user's company

Holes PATCH

This endpoint can be used to Update values of Holes against a Plan or Project.

ENDPOINT

<https://metrixapi.kruxanalytics.com/api/v1/Holes>

Request Body Sample – Non - Directional Hole

```
{
  "holes": [
    {
      "planId": 477,
      "projectId": 1008,
      "holeId": 1234,
      "status": "Complete"
      "holeName": "NewHole1234",
      "plannedHoleName": "NewHole1234",
      "isDirectional": false,
      "plannedDepth": 150,
      "utmZone": "11N",
      "utmEasting": 13,
      "utmNorthing": 2,
```

```

    "elevation": 100,
    "plannedAzimuth": 120,
    "plannedDip": -2,
    "plannedDepthUnit": "meters",
    "elevationUnit": "meters",
    "mineGridEasting": 44,
    "mineGridNorthing": 12,
    "holeTypeId": 43,
    "holeSpecifications": "This is a hole added from the API",
    "tenementId": 15,
    "drillPriority": "High",
    "secondaryStatusID": "26"
  }
]

```

Request Body Sample – Directional Hole

```

{
  "holes": [
    {
      "planId": 477,
      "projectId": 1008,
      "holeId": 1234,
      "status": "Complete"
      "holeName": "directionalHole",
      "plannedHoleName": "NewHole1234",
      "secondaryStatusID": "26",
      "utmEasting": 12.34,
      "utmNorthing": 12.56,
      "utmZone": "1N",
      "elevation": 450,
      "elevationUnit": "feet",
      "mineGridEasting": 10.76,

```

```
        "mineGridNorthing": 3.90,  
        "isDirectional": true,  
        "holeTypeId": 43,  
        "holeSpecifications": "This is a directional hole added from the API",  
        "tenementId": 15,  
        "drillPriority": "High",  
        "directionalUnit": "meters",  
        "targetNorthing": 20,  
        "targetEasting": 40,  
        "targetElevation": 2,  
        "plannedDogleg": 31,  
        "plannedGtf": 4,  
        "startType": "Hole Bottom",  
        "directionalDepthFrom": 0,  
        "directionalDepthTo": 35,  
        "directionalInclinationStart": 0,  
        "directionalInclinationEnd": 44,  
        "directionalAzimuthStart": 20,  
        "directionalAzimuthEnd": 35  
    }  
]  
}
```

Response Body

```
{"InvalidRecords":[],"Status":1}
```

Errors and Status codes

The error and status codes can be referenced in the [APPENDIX](#)

Field Validations

When inserting Holes against a designated Plan or Project, there are specific restrictions governing the values that may be entered into each field. These restrictions are intended to ensure data consistency

and integrity during Hole creation. Please ensure all Hole creation entries strictly adhere to the allowed values for each field. Invalid entries will result in errors or rejected submissions.

Below is the complete list of fields along with their corresponding allowed values:

Fields	Allowed Values
planId	Valid PlanID. This can be retrieved using endpoint https://metrixapi.kruxanalytics.com/api/v1/Plans
Status	This accepts one of three string values: 1. "Cancelled" 2. "Abandoned" 3. "Complete"
projectId	Valid ProjectID. This can be retrieved using endpoint https://metrixapi.kruxanalytics.com/api/v1/Lookups/projects
holeName	Enter a Valid Hole Name that is unique in the Project/Plan the hole is being uploaded to
plannedHoleName	Enter a Valid Hole Name that is unique in the Project/Plan the hole is being uploaded to
isDirectional	This accepts one of two values: 1. true 2. false
plannedDepth	Numerical value, allows decimals
utmZone	Valid UTMZone values, numerical values between 1-60 inclusive appended by N or S. Eg: 1N or 60S
utmEasting	Numerical value, allows decimals
utmNorthing	Numerical value, allows decimals
elevation	Numerical value, allows decimals
plannedAzimuth	Numerical value, allows decimals
plannedDip	Numerical value, allows decimals
plannedDepthUnit	This accepts one of two string values: 1. "Meters" 2. "Feet"
elevationUnit	This accepts one of two string values: 1. "Meters" 2. "Feet"
mineGridEasting	Numerical value, allows decimals
mineGridNorthing	Numerical value, allows decimals
holeTypeId	Valid holeTypeID. This can be retrieved using endpoint https://metrixapi.kruxanalytics.com/api/v1/Holes/types
holeSpecifications	Text field
tenementId	Valid TenementID. This can be retrieved using endpoint https://metrixapi.kruxanalytics.com/api/v1/Lookups/tenements
targetNorthing	Numerical value, allows decimals
targetEasting	Numerical value, allows decimals
targetElevation	Numerical value, allows decimals

directionalUnit	This accepts one of two string values: 1. "Meters" 2. "Feet"
plannedDogleg	Numerical value, allows decimals
plannedGtf	Numerical value, no decimals allowed
startType	Allows one of the four string values: 1. "Hole Bottom" 2. "Cut Curve" 3. "Wedge" 4. "Cement Plug"
directionalDepthFrom	Numerical value, allows decimals
directionalDepthTo	Numerical value, allows decimals
directionalInclinationStart	Numerical value, allows decimals
directionalInclinationEnd	Numerical value, allows decimals
directionalAzimuthStart	Numerical value, allows decimals
directionalAzimuthEnd	Numerical value, allows decimals
drillPriority	Allows one of the four string values: 1. "Critical" 2. "High" 3. "Medium" 4. "Low"
secondaryStatusID	Valid SecondaryStatusID. This can be retrieved using endpoint https://metrixapi.kruxanalytics.com/api/v1/Lookups/secondaryStatuses
holeID	Valid HoleID. This can be retrieved using endpoint https://metrixapi.kruxanalytics.com/api/v1/Holes?isPlan=true

DELETE Methods

These methods are only available to Resource Companies.

Plan DELETE

This endpoint will allow deleting plans if there is no data attached to them. If there is data attached, the plan will not be deleted and corresponding error codes will be returned.

ENDPOINT

<https://metrixapi.kruxanalytics.com/api/v1/Plans>

AUTHORIZATION

This request uses the Bearer Token

HEADERS

None

REQUEST PARAMETER

None



REQUEST BODY

```
{
  "planIds": [
    0,1
  ]
}
```

Response

```
{
  "status": 1
}
```

Holes **DELETE**

This endpoint will allow deleting holes if there is no data attached to them. If there is data attached, the hole will not be deleted and corresponding error codes will be returned.

ENDPOINT

<https://metrixapi.kruxanalytics.com/api/v1/Holes>

AUTHORIZATION

This request uses the Bearer Token

HEADERS

None

REQUEST PARAMETER

None

REQUEST BODY

```
{
  "holeIds": [
    0,1
  ]
}
```

RESPONSE



```
{
  "status": 1
}
```

Export API Suite

This Export API Suite can be used to retrieve Incremental data

Companies GET

<https://metrixapi.kruxanalytics.com/api/v2/Export/Companies>

Used to obtain CompanyIDs for all divisions in user's organization for which the user has permission to export data. CompanyId is a required parameter for GetData call.

AUTHORIZATION

This request uses the Bearer Token

HEADERS

None

Authorization

Bearer <your_token>

RESPONSE

```
{
  "Table": [
    {
      "CompanyID": 12345,
      "CompanyName": "Demo Company",
      "IsParentCompany": 0
    }
  ]
}
```


ExportTypes GET

This is available to both Drilling and Resource companies. It is used to obtain the list of unique queries available in the API that can be used to selectively return required data. "id" from the JSON response must be entered as the **queryName** parameter for the **GetData** call.

ENDPOINT

<https://metrixapi.kruxanalytics.com/api/v2/Export/ExportTypes>

AUTHORIZATION

This request uses the Bearer Token

HEADERS

None

Authorization

Bearer <your_token>

RESPONSE

```
[
  {
    "id": "DSRActivity",
    "description": "Returns all Daily Shift Report activities that have been added-deleted-modified"
  },
  {
    "id": "DSRActivityEquipment",
    "description": "Returns all equipment attached to Daily Shift Report activities that have been added-deleted-modified"
  },
  {
    "id": "DSREquipment",
    "description": "Returns all Daily Shift Report equipment that has been added-deleted-modified"
  },
  {
    "id": "DSRConsumables",
    "description": "Returns all Daily Shift Report consumables that have been added-deleted-modified"
  },
  {
    "id": "TSWorkersLabour",
    "description": "Returns all Timesheet workers that have been added-deleted-modified"
  },
  {

```

```

    "id": "TSActivityLabour",
    "description": "Returns all workers attached to Timesheet activities that have been added-deleted-
modified"
  },
  {
    "id": "TSActivityEquipment",
    "description": "Returns all Equipment attached to Timesheet activities that have been added-deleted-
modified"
  },
  {
    "id": "EquipmentRental",
    "description": "Returns all Equipment Rentals that have been added-deleted-modified"
  },
  {
    "id": "AdditionalCharges",
    "description": "Returns all Additional Charges that have been added-deleted-modified"
  },
  {
    "id": "DSR",
    "description": "Returns all Daily Shift Reports that have been added-deleted-modified"
  },
  {
    "id": "DSRAdditionalCharges",
    "description": "Returns all DSR Additional Charges Rate that have been added/deleted/ modified"
  },
  {
    "id": "TS",
    "description": "Returns all Timesheets that have been added-deleted-modified"
  },
  {
    "id": "DSRFluids",
    "description": "Returns all DSR Fluids that have been added/ deleted / modified"
  },
  {
    "id": "DSRSamples",
    "description": "Returns all DSR Samples that have been added/ deleted / modified"
  },
  {
    "id": "DSRWorkersLabour",
    "description": "Returns all DSR Workers Labour that have been added / deleted / modified"
  },
  {
    "id": "DSRActivityLabour",
    "description": "Returns all DSR Activity that have been added/ deleted / modified"
  },
  {
    "id": "DSRPerDrillRate",

```

```

    "description": "Returns all DSR Per Drill Rate that have been added/ deleted / modified"
  },
  {
    "id": "TSConsumables",
    "description": "Returns all Timesheet consumables that have been added/ deleted /modified"
  },
  {
    "id": "TSFluids",
    "description": "Returns all Timesheet Fluids that have been added/ deleted / modified"
  },
  {
    "id": "Holes",
    "description": "Returns all Holes that have been added / deleted / modified"
  }
  {
    "id": "DSRActivityCodes",
    "description": "Returns all Krux codes for Daily Shift Report activity that have been added-deleted-
modified"
  }
  {
    "id": "DSRCoreRun",
    "description": "Returns all Core Runs for Daily Shift Report that have been added-deleted-modified"
  }
  {
    "id": "DSRDirectionalRun",
    "description": "Returns all Directional Runs for Daily Shift Report that have been added-deleted-
modified"
  }
]

```

ID	Description
DSRActivity	Returns all Daily Shift Report activities that have been added/deleted/modified since exportDateTime
DSRActivityEquipment	Returns all equipment attached to Daily Shift Report activities that have been added/deleted/modified since exportDateTime
DSREquipmentUsage	Returns all Daily Shift Report equipment that has been added/deleted/modified since exportDateTime
DSRConsumables	Returns all Daily Shift Report consumables that have been added/deleted/modified since exportDateTime
TSWorkersLabour	Returns all Timesheet workers that have been added/modified since exportDateTime
TSActivityLabour	Returns all workers attached to Timesheet activities that have been added/modified since exportDateTime
TSActivityEquipment	Returns all workers attached to Timesheet activities that have been added/deleted/modified since exportDateTime
EquipmentRental	Returns all Equipment Rentals that have been added/deleted/modified since exportDateTime

AdditionalCharges	Returns all Additional Charges added/deleted/modified since exportDateTime
DSR	Returns all Daily Shift Reports that have been added/deleted/modified since exportDateTime
TS	Returns all Timesheets that have been added/deleted/modified since exportDateTime
DSRFluids	Returns all Daily Shift Report fluids that have been added/deleted/modified since exportDateTime
Holes	Returns all Holes that have been added/deleted/modified since exportDateTime
DSRSamples	Returns all Daily Shift Report samples that have been added/deleted/modified since exportDateTime
DSRWorkersLabour	Returns all Daily Shift Report workers labour that have been added/deleted/modified since exportDateTime
DSRActivityLabour	Returns all Daily Shift Report activity labour that have been added/modified since exportDateTime
DSRPerDrillRate	Returns all DSR Per Drill Rate that have been added/ modified since exportDateTime
DSRAdditionalCharges	Returns all Additional Charges added/deleted/modified in the DSRs since exportDateTime
TSConsumables	Returns all Timesheet consumables that have been added/ modified since exportDateTime
TSFluids	Returns all Timesheet Fluids that have been added/ modified since exportDateTime
DSRActivityCodes	Returns all Krux unique codes for the DSR Activities that have been added / deleted / modified. The codes include unique Krux level <i>Work SubCategoryID</i> , <i>SubCategoryTypeID</i> , <i>ContractActivityID</i> <i>ContractActivityRateID</i> codes for DSR Activities.
DSRCoreRuns	Returns all Daily Shift Report Core Runs that have been added/deleted/modified since exportDateTime
DSRDirectionalRuns	Returns all Daily Shift Report Directional Runs that have been added/deleted/modified since ExportDateTime

Retrieve Incremental Hole Data

GetData GET

This method is used to pull incremental data out of Krux.

ENDPOINT

<https://metrixapi.kruxanalytics.com/api/v2/Export/GetData/>

Used to export data based on queryName specified in the call. All available options can be retrieved using the ExportTypes call.



AUTHORIZATION

This request uses the Bearer Token

HEADERS

None

Authorization

Bearer <your_token>

REQUEST PARAMETERS

Key	Value	Description
companyId	Integer (12345)	The company identifier used to specify which division of the company data will be returned for. The CompanyID for all divisions in your organization can be obtained by making a request to https://metrixapi.kruxanalytics.com/api/v2/Export/Companies
exportDateTime	Date (2021-01-22 17:42:08.390)	The starting datetime after which data is exported. Any data that has been added, deleted or modified since exportDateTime will be returned.
queryName	String (DSR)	The list of all queries can be obtained by making a request to https://metrixapi.kruxanalytics.com/api/v2/Export/ExportTypes

Errors

Invalid parameters will result in a 400 Bad Requests status. The message keyname in the response json will contain the parameter name with the validation error. The 3 possible messages include:

- Invalid queryName
- Invalid companyId
- Invalid exportDateTime

```
{  
  "$id": "1",
```

```
"message": "Invalid queryName"
}
```

RESPONSE

Each queryName has a unique json response.

Working with the API Data

UIDs

Every data record returned by the API has a unique identifier (UID). Each UID starts with an integer that identifies what type of data it is and then includes one to three additional IDs required to ensure uniqueness of the UID. The general format is of the UID is

Data Type – Report Record ID – Charge ID – Context Dependent ID

Data Type – An integer that identifies what type of data the record is. Included in all UIDs.

Report Record ID – Unique identifier for the report record entered by the user (e.g. A drilling activity record from a Daily Shift Report). Included in all UIDs.

Charge ID – Unique identifier for the charges calculated based on an applicable rate in the contract. (e.g. Charges calculated based on a meterage rate for a drilling activity). Only included for data that can have associated contract rates.

Context Dependent ID – Will be different for each type of data. Only included when required for uniqueness.

The format for each individual UID is included in the following sections.

Processing Incremental Data

All data returned by the GetData method is incremental data based on the exportDateTime parameter provided. This means that only data that is new or has changed in some way since the provided exportDateTime will be included. If any of the following have occurred since exportDateTime the record will be included:

- DSR/TS
 - Approved for contracts with Report Approval Required = Yes
 - Validated for contracts with Report Approval Required = No
 - Rejected
 - Deleted
 - Header data was modified
 - Individual record within the report was added, modified, or deleted
 - Applicable contract rate was added, modified, or deleted
- Additional Charge/Equipment Rental is approved, modified, or deleted
- Hole Records was added, modified, or deleted

Records with **DeletedFlag** = 'N' represent approved data non-deleted data.

On the other hand, **DeletedFlag** = 'Y' can indicate one of the following scenarios:

1. **New Data:** Data that is not yet approved and is currently pending approval.
2. **Rejected Data:** Data that was rejected by an approver and has reverted to a pending status.
3. **Deleted Data:** Records removed from various sources, such as the Daily Shift Report or Timesheet and rate records deleted from contracts (which will in turn delete associated charges on DSR, TS, etc.). So, in order to distinguish actual deleted data from new or rejected data, we introduced the **DataDeleted** flag.

To process the new incremental data against existing dataset, use the following logic:

UID Exist in Dataset	Delete Flag	DataDeleted	Action Required
Y	Y	Y	Remove record from dataset
Y	Y	N	Remove record in dataset
Y	N	N	Update record in dataset
N	Y	Y	Disregard, do not add record to dataset
N	Y	N	Disregard, do not add record to dataset
N	N	N	Add record to dataset

Using Report and Charge Data

Each type of data returned by the API consists of just report data or report and charge data from an associated contract rate. If a data type also includes charge data, its UID will include a charge ID. The only exception to this is Additional Charges, as they are charge records in and of themselves; they have no associated contract rates. In general, the fields that are associated with the charge portion of a records are:

- Billing Type
- Total Charges
- Currency Code

The other data fields are associated with the report portion of the record.

Each individual report record can have multiple associated contract rates. This means that one report record can return multiple unique records in the API with different charges.

Example - A DSR drilling activity with a meterage rate and an hourly rate. This will return 2 records in the API.

Data Field	Record 1	Record 2
UID	1-12345-333	1-12345-777
DailyReportID	16556	16556
HoleID	47478	47478
Hole	TYC-1209	TYC-1210
Activity	Drilling	Drilling




Type	Coring	Coring
Bit Size	NQ	NQ
DistanceDrilledFrom	53	53
DistanceDrilledTo	75	75
Billable	Yes	Yes
ActivityHours	4	4
TotalManHours	8	8
Penetration	22	22
BillingType	Per Distance	Per Activity Hour
DistanceFromToUnitAbbr	m	m
TotalCharges	1430	200
CurrencyCode	CAD	CAD

Depending how you want to manipulate the API data you may have to breakout the UID in its separate IDs.

Example Calculation 1 - If you want to find the total of all charges for report record, you have to sum the TotalCharges field for all records that have the same Report Record ID.

Example Calculation 2 – If you want to find total activity hours for a Daily Shift Report. You first have to group the activity records so that you have only one record per Report Record ID. Then sum the Activity Hours field for all activities with the same DailyReportID.

Hole Data

	UID	VARCHAR(22)
	HoleID	INT
	HoleName	NVARCHAR(100)
	HoleStatus	VARCHAR(10)
	CompleteDateTime	DATE
	HoleType	NVARCHAR(200)
	ContractorCompany	VARCHAR(200)
	ContracteeCompany	VARCHAR(200)
	Contract	NVARCHAR(100)
	Project	NVARCHAR(100)
	Plan	NVARCHAR(200)
	FirstActivityDate	DATE
	LastActivityDate	DATE
	MaxDepth	DECIMAL(22, 2)
	PlannedDepth	DECIMAL(22, 2)
	DepthUnit	NVARCHAR(100)
	TotalDistanceDrilled	DECIMAL(22, 3)
	TotalActivityHours	DECIMAL(22, 3)
	TotalDrillingHours	DECIMAL(22, 3)
	Penetration	DECIMAL(22, 3)
	Easting	FLOAT
	Northing	FLOAT
	UTMZone	NVARCHAR(50)
	MineGridEasting	FLOAT
	MineGridNorthing	FLOAT
	Elevation	FLOAT
	ElevationUnit	NVARCHAR(100)
	PlannedAzimuth	FLOAT
	PlannedDip	FLOAT
	DeletedFlag	VARCHAR(1)
	ExportDateTime	DATETIME
	ContractID	INT

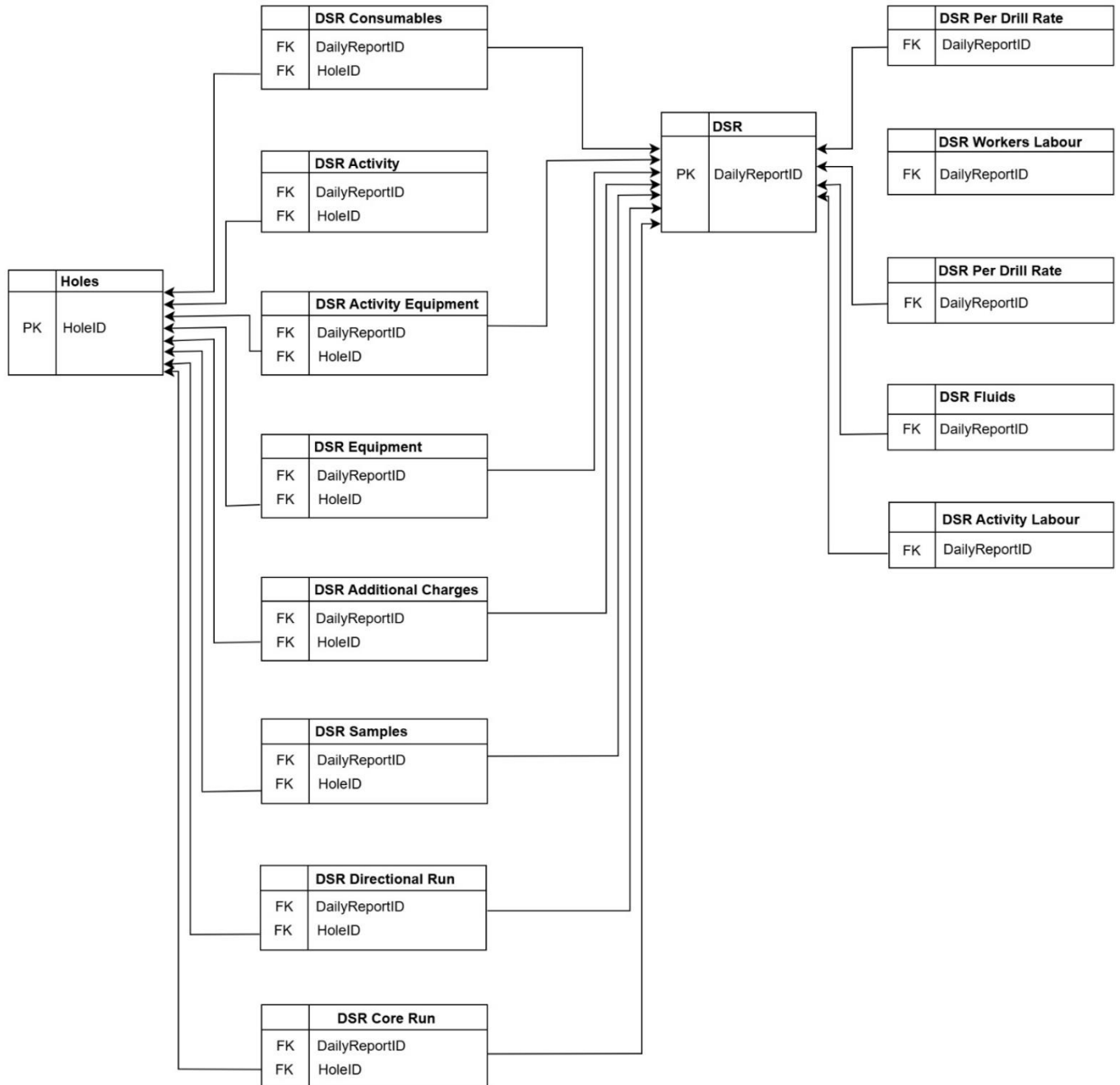
Hole UID Format

14-[HoleID]

The **HoleID** column is the unique identifying number for a hole. Any record associated directly with a hole will include HoleID as well as Hole. Hole is the name for that hole entered by the user and displayed to identify the hole for the users in the KruxMetrix and KruxLog. Hole is very often unique but it is not guaranteed to be unique; for the purposes of processing the data from the API, HoleID must be used.

Daily Shift Report (DSR) Data

DSR Relationship diagram



DSR Data Definitions

DSR UID Format

11-[DailyReportID]



The **DailyReportID** column is the unique identifying number for the Daily Shift Report (DSR). All associated data for the same DSR will have the same DailyReportID.

DSR

UID	VARCHAR(22)
DailyReportID	BIGINT
ContractorCompany	VARCHAR(200)
ContracteeCompany	VARCHAR(200)
Contract	NVARCHAR(100)
Project	NVARCHAR(100)
Drill	NVARCHAR(100)
ReportDate	DATE
Status	VARCHAR(30)
Supervisor	NVARCHAR(201)
Shift	NVARCHAR(100)
ValidatedBy	NVARCHAR(201)
ValidatedDate	DATETIME
ApprovedBy	NVARCHAR(201)
ApprovedDate	DATETIME
DeletedFlag	VARCHAR(1)
ExportDateTime	DATETIME
ContractID	INT

DSRWorkerLabour UID Format

16-[DSRWorkerLabourID]-[DSRWorkerLabourChargeID]

The DSRWorkerLabourID is the unique identifier for the workers with payroll hours entered in the DSR. The DSRWorkerLabourChargeID is the unique identifier for the charges calculated for that worker based on an applicable rate in the contract. If there is no applicable rate in the contract, DSRWorkerLabourChargeID will be null.

DSR Workers Labour

	UID	VARCHAR(42)
	DailyReportID	BIGINT
	Name	NVARCHAR(201)
	Role	NVARCHAR(50)
	PayrollHours	DECIMAL(6, 3)
	BillingType	VARCHAR(100)
	TotalCharges	DECIMAL(22, 2)
	CurrencyCode	VARCHAR(100)
	DeletedFlag	VARCHAR(1)
	ExportDateTime	DATETIME
	DataDeleted	VARCHAR(1)

DSRActivity UID Format

1-[DSRActivityID]-[DSRActivityChargeID]

The DSRActivityID is the unique identifier for the activity entered in the DSR. The DSRActivityChargeID is the unique identifier for the charges calculated for that activity based on an applicable rate in the contract. If there is no applicable rate in the contract, DSRActivityChargeID will be null.

DSR Activity

	UID	VARCHAR(42)
	DailyReportID	BIGINT
	HoleID	INT
	Hole	NVARCHAR(100)
	Activity	VARCHAR(100)
	Type	NVARCHAR(200)
	BitSize	VARCHAR(100)
	DistanceDrilledFrom	DECIMAL(22, 3)
	DistanceDrilledTo	DECIMAL(22, 3)
	Distance	DECIMAL(25, 3)
	Depth	DECIMAL(22, 2)
	Billable	VARCHAR(1)
	ActivityHours	DECIMAL(7, 3)
	TotalManHours	DECIMAL(22, 3)
	Penetration	DECIMAL(22, 3)
	BillingType	VARCHAR(100)
	DistanceFromToUnitAbbr	NVARCHAR(50)
	DistanceUnitAbbr	NVARCHAR(50)
	DepthUnitAbbr	NVARCHAR(50)
	TotalCharges	DECIMAL(22, 2)
	CurrencyCode	VARCHAR(100)
	DeletedFlag	VARCHAR(1)
	ExportDateTime	DATETIME
	WorkSubCategoryID	INT
	WorkSubCategoryTypeID	INT
	DataDeleted	VARCHAR(1)
	ChargeFrom	FLOAT
	ChargeTo	FLOAT
	Comments	VARCHAR(42)
	BitSizeID	INT

DSRActivityEquipment UID Format

3-[DSRActivityEquipmentID]-[DSRActivityEquipmentChargeID]

The DSRActivityEquipmentID is the unique identifier for the equipment entered for an activity in the DSR. The DSRActivityEquipmentChargeID is the unique identifier for the charges calculated for that equipment based on an applicable rate in the contract. If there is no applicable rate in the contract, DSRActivityEquipmentChargeID will be null.

DSR Activity Equipment

	UID	VARCHAR(42)
	DailyReportID	BIGINT
	HoleID	INT
	Hole	NVARCHAR(100)
	Activity	VARCHAR(100)
	Equipment	NVARCHAR(100)
	EquipmentHours	DECIMAL(6, 3)
	EquipmentUnit	NVARCHAR(30)
	BillingType	VARCHAR(100)
	TotalCharges	DECIMAL(22, 2)
	CurrencyCode	VARCHAR(100)
	DeletedFlag	VARCHAR(1)
	ExportDateTime	DATETIME
	ContractEquipmentID	INT
	DataDeleted	VARCHAR(1)

DSRActivityLabour UID Format

17-[DSRActivityLabourID]-[DSRActivityLabourChargeID]

The DSRActivityLabourID is the unique identifier for the workers with man hours entered for an activity in the DSR. The DSRActivityLabourChargeID is the unique identifier for the charges calculated for that worker based on an applicable rate in the contract. If there is no applicable rate in the contract, DSRActivityLabourChargeID will be null.

DSR Activity Labour

UID	VARCHAR(42)
DailyReportID	BIGINT
Activity	VARCHAR(100)
Name	NVARCHAR(201)
Role	NVARCHAR(50)
ManHours	DECIMAL(6, 3)
BillingType	VARCHAR(100)
TotalCharges	DECIMAL(22, 2)
CurrencyCode	VARCHAR(100)
DeletedFlag	VARCHAR(1)
ExportDateTime	DATETIME
DataDeleted	VARCHAR(1)

DSREquipmentUsage UID Format

4-[DSREquipmentID]-[DSREquipmentChargeID]]-[EquipmentUnitID]

The DSREquipmentID is the unique identifier for the equipment record entered in the DSR. The EquipmentUnitID is the unique identifier for a specific equipment unit that was included in the equipment record. If no specific equipment unit is selected, EquipmentUnitID will be null. The DSREquipmentChargeID is the unique identifier for the charges calculated for that equipment based on an applicable rate in the contract. If there is no applicable rate in the contract, DSREquipmentChargeID will be null.

DSR Equipment

 UID	VARCHAR(62)
DailyReportID	BIGINT
HoleID	INT
Hole	NVARCHAR(100)
Equipment	NVARCHAR(100)
EquipmentUnit	NVARCHAR(30)
Quantity	DECIMAL(22, 3)
BillingType	VARCHAR(100)
TotalCharges	DECIMAL(22, 2)
CurrencyCode	VARCHAR(100)
DeletedFlag	VARCHAR(1)
ExportDateTime	DATETIME
ContractEquipmentID	INT
DataDeleted	VARCHAR(1)

DSRConsumables UID Format

5-[DSRConsumablesID]-[DSRConsumablesChargeID]

The DSRConsumablesID is the unique identifier for the consumable entered in the DSR. The DSRConsumablesChargeID is the unique identifier for the charges calculated for that consumable based on an applicable rate in the contract. If there is no applicable rate in the contract, DSRConsumablesChargeID will be null.

DSR Consumables

 UID	VARCHAR(42)
DailyReportID	BIGINT
HoleID	INT
Hole	NVARCHAR(100)
Consumable	NVARCHAR(100)
Quantity	DECIMAL(22, 3)
TotalCharges	DECIMAL(22, 2)
CurrencyCode	VARCHAR(100)
DeletedFlag	VARCHAR(1)
ExportDateTime	DATETIME
ContractConsumablesRateID	INT
DataDeleted	VARCHAR(1)

DSRFluids UID Format

13-[DSRFluidsID]-[DSRFluidsChargeID]



The DSRFluidsID is the unique identifier for the fluid volume entered in the DSR. The DSRFluidsChargeID is the unique identifier for the charges calculated for that fluid based on an applicable rate in the contract. If there is no applicable rate in the contract, DSRFluidsChargeID will be null.

DSR Fluids

UID	VARCHAR(42)
DailyReportID	BIGINT
Fluid	VARCHAR(500)
FluidType	NVARCHAR(500)
Volume	FLOAT
Unit	NVARCHAR(100)
CostPerUnit	DECIMAL(11, 2)
Markup	DECIMAL(6, 2)
TotalCharges	DECIMAL(22, 2)
CurrencyCode	VARCHAR(100)
DeletedFlag	VARCHAR(1)
ExportDateTime	DATETIME
DataDeleted	VARCHAR(1)

DSRSamples UID Format

15-[DSRSamplesID]

The DSRSamplesID is the unique identifier for the sample entered in the DSR.

DSR Samples

 UID	VARCHAR(22)
DailyReportID	BIGINT
HoleID	INT
Hole	NVARCHAR(100)
SampleType	VARCHAR(100)
From	FLOAT
To	FLOAT
Unit	NVARCHAR(100)
DeletedFlag	VARCHAR(1)
ExportDateTime	DATETIME
DataDeleted	VARCHAR(1)

DSRPerDrillRate UID Format

18-[DSRPerDrillRateID]

The DSRPerDrillRateID is the unique identifier for the Per Drill Rate entered in the DSR.

DSR Per Drill Rate

 UID	VARCHAR(22)
DailyReportID	BIGINT
BillingType	VARCHAR(513)
TotalCharges	DECIMAL(22, 2)
CurrencyCode	VARCHAR(100)
Drill	NVARCHAR(100)
DeletedFlag	VARCHAR(1)
ExportDateTime	DATETIME

DSRAdditionalCharges UID Format

10-[DSRAdditionalChargeID]

The DSRAdditionalChargeID is the unique identifier for the DSR additional charge record.

DSR Additional Charge

 UID	VARCHAR(22)
DailyReportID	BIGINT
HoleID	INT
HoleName	NVARCHAR(100)
Description	NVARCHAR(500)
ChargeRate	DECIMAL(16, 3)
ChargeableQuantity	FLOAT
TotalCharges	DECIMAL(22, 2)
CurrencyCode	VARCHAR(100)
DeleteFlag	VARCHAR(1)
ExportDateTime	DATETIME
DataDeleted	VARCHAR(1)

DSRActivityCodes UID Format

22-[DSRActivityID]-[DSRActivityChargeID]

The DSRActivityID is the unique identifier for the activity entered in the DSR. The DSRActivityChargeID is the unique identifier for the charges calculated for that activity based on an applicable rate in the contract. If there is no applicable rate in the contract, DSRActivityChargeID will be null.

DSR Activity Code Relationship diagram



DSR Activity Codes

	UID	VARCHAR(42)
	DSRActivityUID	VARCHAR(41)
	Activity	VARCHAR(100)
	WorkSubCategoryID	INT
	Type	NVARCHAR(200)
	WorkSubCategoryTypeID	INT
	ContractName	NVARCHAR(100)
	ContractActivityID	BIGINT
	ContractActivityRateID	BIGINT
	DistanceFrom	DECIMAL(22, 3)
	DistanceTo	DECIMAL(22, 3)
	DipFrom	DECIMAL(20, 2)
	DipTo	DECIMAL(20, 2)
	TimeFrom	INT
	TimeTo	INT
	DefinedRateName	NVARCHAR(100)
	DefinedRate	MONEY
	ActivityBillingType	VARCHAR(100)
	RateType	VARCHAR(100)
	CustomRate	MONEY
	DistanceFromToUnit	NVARCHAR(50)
	CreatedDateTime	DATETIME
	LastModDateTime	DATETIME
	LastModByUserName	VARCHAR(255)
	ExportDateTime	DATETIME
	ContractID	INT


DSRCoreRun UID Format

23-[DailyReportHoleActivityCoreRunID]



The *DailyReportHoleActivityCoreRunID* is the unique identifier for the core run entered for a drilling activity in the DSR.

Core Runs


	UID	VARCHAR(19)
	DailyReportID	BIGINT
	HoleID	INT
	CoreNumber	BIGINT
	CoreBarrelSize	FLOAT
	ConstantStickUp	FLOAT
	DepthFrom	DECIMAL(22, 2)
	DepthTo	DECIMAL(22, 2)
	TotalDrilled	DECIMAL(22, 2)
	Recovered	DECIMAL(22, 2)
	CoreLoss	DECIMAL(22, 2)
	Recovered%	DECIMAL(22, 2)
	Orientation	VARCHAR(3)
	StickUp	FLOAT
	CoreBoxTop	NVARCHAR(255)
	CoreBoxBottom	NVARCHAR(255)
	DeletedFlag	VARCHAR(1)

DSRDirectionalRun UID Format

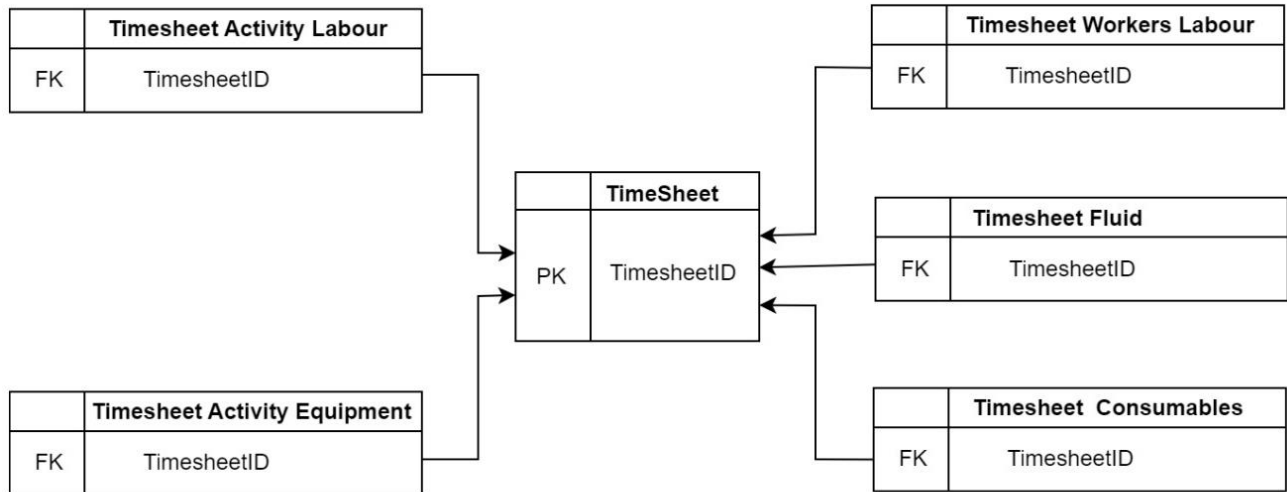
23-[*DailyReportHoleActivityDirectionalRunID*]

The *DailyReportHoleActivityCoreRunID* is the unique identifier for the core run entered for a drilling activity in the DSR.

Directional Run

	UID	VARCHAR(19)
	DailyReportID	BIGINT
	HoleID	INT
	DepthFrom	NVARCHAR(75)
	DepthTo	NVARCHAR(75)
	StartTime	CHAR(5)
	EndTime	CHAR(5)
	PumpingDuration	CHAR(5)
	PullingDuration	CHAR(5)
	DrillingDuration	CHAR(5)
	RateOfPenetration	DECIMAL(22, 1)
	WOB	NVARCHAR(75)
	Torque	NVARCHAR(75)
	RPM	DECIMAL(22, 1)
	WaterFlow	NVARCHAR(60)
	WaterPressure	NVARCHAR(60)
	DoglegSetting	DECIMAL(22, 1)
	DialIndication	NVARCHAR(75)
	CCG	DECIMAL(22, 1)
	ToolfaceBD	INT
	ToolfaceAD	INT
	InclinationBD	DECIMAL(22, 1)
	InclinationAD	DECIMAL(22, 1)
	EndReason	NVARCHAR(30)
	DeletedFlag	VARCHAR(1)

TS Relationship diagram



TS UID Format

12-[TimesheetID]

The **TimesheetID** column is the unique identifying number for the Timesheet (TS). All associated data for the same timesheet will have the same TimesheetID.

TS

	UID	VARCHAR(22)
	TimesheetID	INT
	ContractorCompany	VARCHAR(200)
	ContracteeCompany	VARCHAR(200)
	Contract	NVARCHAR(100)
	Project	NVARCHAR(100)
	ReportDate	DATE
	Status	VARCHAR(30)
	SubmittedBy	NVARCHAR(201)
	ValidatedBy	NVARCHAR(201)
	ValidatedDate	DATETIME
	ApprovedBy	NVARCHAR(201)
	ApprovedDate	DATETIME
	DeletedFlag	VARCHAR(1)
	ExportDateTime	DATETIME
	ContractID	INT

TSWorkersLabour UID Format

6-[TSWorkerLabourID]-[TSWorkerLabourChargeID]

The TSWorkerLabourID is the unique identifier for the workers with payroll hours entered in the TS. The TSWorkerLabourChargeID is the unique identifier for the charges calculated for that worker based on an applicable rate in the contract. If there is no applicable rate in the contract, TSWorkerLabourChargeID will be null.

TS Workers Labour


 UID	VARCHAR(42)
TimesheetID	INT
Name	NVARCHAR(201)
Role	NVARCHAR(50)
PayrollHours	DECIMAL(6, 3)
BillingType	VARCHAR(100)
TotalCharges	DECIMAL(22, 2)
CurrencyCode	VARCHAR(100)
DeletedFlag	VARCHAR(1)
ExportDateTime	DATETIME
DataDeleted	VARCHAR(1)

TSActivityLabour UID Format

7-[TSActivityLabourID]-[TSActivityLabourChargeID]

The TSActivityLabourID is the unique identifier for the workers with man hours entered for an activity in the TS. The TSActivityLabourChargeID is the unique identifier for the charges calculated for that worker based on an applicable rate in the contract. If there is no applicable rate in the contract, TSActivityLabourChargeID will be null.

TS Activity Labour

 UID	VARCHAR(42)
TimesheetID	INT
Name	NVARCHAR(201)
Role	NVARCHAR(50)
Activity	NVARCHAR(100)
ManHours	DECIMAL(6, 3)
BillingType	VARCHAR(100)
TotalCharges	DECIMAL(22, 2)
CurrencyCode	VARCHAR(100)
DeletedFlag	VARCHAR(1)
ExportDateTime	DATETIME
DataDeleted	VARCHAR(1)

TSActivityEquipment UID Format

8-[TSActivityEquipmentID]-[TSActivityEquipmentChargeID]

The TSActivityEquipmentID is the unique identifier for the equipment entered for an activity in the TS. The TSActivityEquipmentChargeID is the unique identifier for the charges calculated for that equipment based on an applicable rate in the contract. If there is no applicable rate in the contract, TSActivityEquipmentChargeID will be null.

TS Activity Equipment


 UID	VARCHAR(42)
TimesheetID	INT
Activity	NVARCHAR(100)
Equipment	NVARCHAR(100)
EquipmentHours	DECIMAL(6, 3)
EquipmentUnit	NVARCHAR(30)
BillingType	VARCHAR(100)
TotalCharges	DECIMAL(22, 2)
CurrencyCode	VARCHAR(100)
DeletedFlag	VARCHAR(1)
ExportDateTime	DATETIME
DataDeleted	VARCHAR(1)

TSConsumables UID Format

20-[TSConsumablesID]

The TSConsumablesID is the unique identifier for the timesheet consumables entered for an activity in the TS.


TS Consumables

 UID	VARCHAR(42)
TimesheetID	INT
TimesheetConsumableID	BIGINT
Consumables	NVARCHAR(100)
Quantity	FLOAT
TotalCharges	DECIMAL(22, 2)
CurrencyCode	VARCHAR(100)
ExportDeleteFlag	VARCHAR(1)
ExportDateTime	DATETIME
DataDeleted	VARCHAR(1)

TSFluid UID Format*19-[TSFluid]*

The TimesheetFluidConsumptionID is the unique identifier for the timesheet fluid consumables entered for an activity in the TS.

TS Fluid

 UID	VARCHAR(22)
TimesheetID	INT
TimesheetFluidConsumptionID	BIGINT
Fluid	VARCHAR(500)
FluidType	NVARCHAR(500)
Volume	FLOAT
Unit	NVARCHAR(100)
CostPerUnit	DECIMAL(11, 2)
Markup	DECIMAL(6, 2)
TotalCharges	DECIMAL(38, 2)
CurrencyCode	VARCHAR(100)
ExportDeleteFlag	VARCHAR(1)
ExportDateTime	DATETIME
DataDeleted	VARCHAR(1)

Equipment Rentals Data

 UID	VARCHAR(42)
ContractorCompany	VARCHAR(200)
ContracteeCompany	VARCHAR(200)
Contract	NVARCHAR(100)
Project	NVARCHAR(100)
Drill	NVARCHAR(100)
Equipment	NVARCHAR(100)
RentalStartDate	DATE
RentalEndDate	DATE
Quantity	INT
BillingType	VARCHAR(100)
TotalCharges	DECIMAL(22, 2)
CurrencyCode	VARCHAR(100)
ClientApprovedBy	NVARCHAR(201)
ClientApprovedDate	DATETIME
ContractorApprovedBy	NVARCHAR(201)
ContractorApprovedDate	DATETIME
DeletedFlag	VARCHAR(1)
ExportDateTime	DATETIME
DataDeleted	VARCHAR(1)
ContractID	INT

EquipmentRental UID Format

9-[EquipmentRentalID]-[ContractEquipmentID]

The EquipmentRentalID is the unique identifier for the equipment rental record. The ContractEquipmentID is the unique identifier for the charges calculated for that equipment rental based on an applicable rate in the contract. If there is no applicable rate in the contract, EquipmentRentalChargeID will be null.

Additional Charges Data

AdditionalCharges UID Format


10-[AdditionalChargeID]



The AdditionalChargeID is the unique identifier for the additional charge record.



Additional Charges

	UID	VARCHAR(22)
	ContractorCompany	VARCHAR(200)
	ContracteeCompany	VARCHAR(200)
	Contract	NVARCHAR(100)
	Project	NVARCHAR(100)
	Drill	NVARCHAR(100)
	HoleID	INT
	Hole	NVARCHAR(100)
	ChargeDate	DATE
	Description	NVARCHAR(500)
	ChargeRate	DECIMAL(22, 2)
	ChargeableQuantity	DECIMAL(22, 2)
	TotalCharges	DECIMAL(22, 2)
	CurrencyCode	VARCHAR(100)
	ClientApprovedBy	NVARCHAR(201)
	ClientApprovedDate	DATETIME
	ContractorApprovedBy	NVARCHAR(201)
	ContractorApprovedDate	DATETIME
	DeletedFlag	VARCHAR(1)
	ExportDateTime	DATETIME
	DataDeleted	VARCHAR(1)
	ContractID	INT

Appendix

Status Codes

Status Code	Status
1	All records succeeded
2	Warning for records that failed while some records succeeded
3	All records failed with error codes for failed records

Error Codes

ImportType	FailureCode	FailureText
All	0	Successful
Plan	1	Duplicate Plan Name
Plan	2	LocationID does not exist
Plan	3	CurrencyID does not exist
Plan	4	PlannedDepthUnitID does not exist
Plan	5	PlanTypeID does not exist
Plan	6	PlanID does not exist
Plan	7	Budget is not null but CurrencyID is null
Plan	8	PlannedDepth is not null but PlannedDepthUnitID is null
Hole	101	Duplicate Hole Name
Hole	102	Duplicate Planned Hole Name
Hole	103	DrillPriority does not exist
Hole	104	ElevationUnitID does not exist
Hole	105	HoleTypeID does not exist
Hole	106	IsDirectional value does not exist
Hole	107	PlanID does not exist
Hole	108	PlannedDepthUnitID does not exist
Hole	109	ProjectID does not exist
Hole	110	StartType does not exist
Hole	111	TenementID does not exist
Hole	112	UTMZoneID does not exist
Hole	113	PlannedDepth is not null but PlannedDepthUnitID is null
Hole	114	ElevationUnit is required when Elevation has a value
Hole	115	Additional Charges exist
Hole	116	DSR Activities exist
Hole	117	DSR Additional Charges exist
Hole	118	DSR Consumables exist
Hole	119	DSR Equipment Usage exists
Hole	120	DSR Samples exist
Hole	121	SecondaryStatus does not exist
Hole	122	HoleID does not exist
Hole	123	Hole is already deleted

Hole	124	SecondaryStatus is not correct
Plan	200	Empty plan name.
Plan	201	Currency id needed when budget has a value.
Plan	202	PlannedDepthUnit is needed when PlannedDepth has a value.
Plan	203	Plan name is too long.
Plan	204	Duplicate plan name in the data set.
Plan	205	Invalid plan id.
Plan	206	Invalid plan status.
Plan	207	Invalid plan name.
Plan	208	Duplicate plan id in the data set.
Plan	209	Plan not found.
Plan	210	Plan has hole(s) attached.
Hole	300	Empty Planned hole name.
Hole	301	Hole name or Planned hole name is required to add a hole.
Hole	302	PlannedDepthUnit is needed when PlannedDepth has a value.
Hole	303	Hole name is too long.
Hole	304	HoleSpecifications is too long.
Hole	305	Plan id or Project id is required to add a hole.
Hole	306	StartType has an invalid value.
Hole	307	DrillPriority has an invalid value.
Hole	308	Status has an invalid value.
Hole	309	Invalid hole id.
Hole	310	Hole id is required to update hole.
Hole	311	Elevation unit id has an invalid value.
Hole	312	Duplicate hole name in the data set.
Hole	313	Invalid Project id.
Hole	314	Duplicate hole id in the data set.
Hole	315	ElevationUnit is required when Elevation has a value.
Hole	316	ElevationUnit has an invalid value.
Hole	317	Invalid Utm Zone.
Hole	318	UtmEasting has value without UTM zone.
Hole	319	UtmNorthing has value without UTM zone.
Hole	400	TargetNorthing is only applicable when IsDirectional is true.
Hole	401	TargetEasting is only applicable when IsDirectional is true.
Hole	402	TargetElevation is only applicable when IsDirectional is true.
Hole	403	DirectionalUnit is only applicable when IsDirectional is true.
Hole	404	PlannedDogleg is only applicable when IsDirectional is true.
Hole	405	PlannedGTF is only applicable when IsDirectional is true.
Hole	406	StartType is only applicable when IsDirectional is true.
Hole	407	DirectionalDepthFrom is only applicable when IsDirectional is true.
Hole	408	DirectionalDepthTo is only applicable when IsDirectional is true.
Hole	409	PlannedDepth is only applicable when IsDirectional is false.
Hole	410	PlannedDepthUnit is only applicable when IsDirectional is false.

Hole	411	PlannedAzimuth is only applicable when IsDirectional is false.
Hole	412	PlannedDip is only applicable when IsDirectional is false.
Hole	413	DirectionalInclinationStart is only applicable when IsDirectional is true.
Hole	414	DirectionalInclinationEnd is only applicable when IsDirectional is true.
Hole	415	DirectionalAzimuthStart is only applicable when IsDirectional is true.
Hole	416	DirectionalAzimuthEnd is only applicable when IsDirectional is true.
All	998	SQL Locking issue
All	999	Unknown SQL error