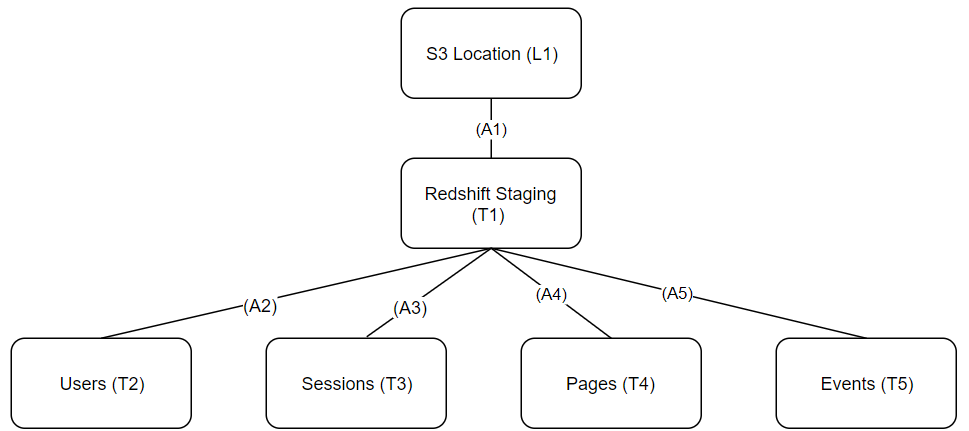
Management of Data in the Raw data transfers and normalization using Amazon Data Pipeline and Redshift

TABLE OF CONTENTS:

1. Processing raw files to staging (A1, T1)
2. Normalizing and Distributing data
   1. Users (A2, T2)
   2. Sessions (A3, T3)
   3. Pages (A4, T4)
   4. Events (A5, T5)



1. Processing raw data to staging:

Staging Table (events\_staging):

|  |  |  |  |
| --- | --- | --- | --- |
| **PK** | **Name** | **Data Type** | **Description** |
| X | id | varchar(150) | DitR unique identifier |
|  | sessionid | varchar(150) | DitR session identifier |
|  | url | varchar(65535) | Full URL of webpage |
|  | urlquery | varchar(65535) | Query string of URL |
|  | type | varchar(150) | click, blur, scroll, pageload |
|  | time | varchar(150) | Client time of event |
|  | eventnumber | integer | Sequential order of events in session |
|  | browser | varchar(150) | Client browser |
|  | target | varchar(65535) | Target of an element that is the focus of a click |
|  | sourceutm | varchar(2000) | Session source based on UTM parameters |
|  | mediumutm | varchar(2000) | Session medium based on UTM parameters |
|  | campaignutm | varchar(2000) | Session campaign based on UTM parameters |
|  | termutm | varchar(2000) | Session term based on UTM parameters |
|  | contentutm | varchar(2000) | Session content based on UTM parameters |
|  | screenresolution | varchar(150) | Screen resolution of client machine |
|  | operatingsystem | varchar(150) | Operating system of client machine |
|  | scrollpercent | float | Percent scroll depth on browser page |
|  | referrer | varchar(2000) | Referrer of session |
|  | windowheight | Integer | Height of browser window |
|  | windowwidth | integer | Width of browser window |
|  | machineid | varchar(150) | Unique ID set on a client browser that persists until cleared |
|  | user | varchar(2000) | Unique ID set with a set user method from site owner |
|  | href | varchar(2000) | Href of a registered click when present |
|  | element | varchar(65535) | Element that is the focus of a click |
|  | xcoordinate | float | X coordinate of click |
|  | ycoordinate | float | Y coordinate of click |
|  | ip | varchar(150) | IP address of client machine |
|  | country | varchar(150) | Country of client based on IP location |
|  | region | varchar(150) | Region of client based on IP location |
|  | city | varchar(150) | City of client based on IP location |
|  | zip | varchar(150) | Zip code of client based on IP location |
|  | timezone | varchar(150) | Time zone of client based on IP location |
|  | latitude | varchar(150) | Latitude of client based on IP location |
|  | longitude | varchar(150) | Longitude of client based on IP location |
|  | createdat | varchar(150) | DitR server timestamp of event |
|  | updatedat | varchar(150) | DitR server timestamp of update |
|  | siteid | Varchar(150) | Unique identifier provided by DitR per website integration |
|  | pagelabel | varchar(150) | Page label set in the DitR tracking by user website |
|  | pagetype | varchar(150) | Page type set in the DitR tracking by user website |
|  | device | varchar(150) | Device type of client (Mobile, Tablet, Desktop) |

AWS Datapipline Configuration:

Activities:

|  |  |
| --- | --- |
| **Field** | **Values** |
| Name | Clear\_Staging |
| Type | SqlActivity |
| Database | RedshiftCluster |
| Runs On | Ec2Instance |
| Script | delete from events\_staging; |

|  |  |
| --- | --- |
| **Field** | **Value** |
| Name | JSON\_to\_Staging |
| Type | SqlActivity |
| Database | RedshiftCluster |
| Runs On | Ec2Instance |
| Script | copy events\_staging  from '#{myInputS3Loc}'  ACCESS\_KEY\_ID 'Your Access Key'  SECRET\_ACCESS\_KEY 'Your Secret Key'  Json 's3://<bucket>/dirt\_jsonpath.json'  maxerror as 250; |
| Depends On | Clear\_Staging |

JSON File:

You will need to create and store the .json file in a S3 location that will always be available. The required JSON file format can be found here. <https://github.com/DataInTheRaw/Amazon_DataPipeline_Redshift_Data_Processing_Normalization/blob/master/ditr_jsonpath.json>

DataNodes:

|  |  |
| --- | --- |
| **Field** | **Value** |
| Name | DestRedshiftTable |
| Type | RedshiftDataNode |
| Database | RedshiftCluster |
| Create Table Sql | #{myRedshiftCreateTableSql} |
| Primary Keys | #{myPrimaryKeys} |
| Table Name | #{myRedshiftTableName} |

|  |  |
| --- | --- |
| **Field** | **Value** |
| Name | S3InputDataNode |
| Type | S3DataNode |
| Directory Path | #{myInputS3Loc} |

Resources:

|  |  |
| --- | --- |
| **Field** | **Value** |
| Name | Ec2Instance |
| Type | Ec2Resource |
| Instance Type | t1.micro |
| Security Groups | #{myRedshiftSecurityGrps} |
| Terminate After | X Hours (2 Hours) |

Other:

|  |  |
| --- | --- |
| **Field** | **Value** |
| Name | RedshiftCluster |
| Type | RedshiftDatabase |
| Connection String | #{myRedshiftJdbcConnectStr} |
| Database Name | #{myRedshiftDbName} |
| Password | #{\*myRedshiftPassword} |
| Username | #{myRedshiftUsername} |

|  |  |
| --- | --- |
| **Field** | **Value** |
| Name | Default |
| Failure and Rerun Mode | CASCADE |
| Resource Role | DataPipelineDefaultResource |
| Role | DataPiplineDefaultRole |
| Schedule Type | ONDEMAND |

Parameters:

|  |  |
| --- | --- |
| **Field** | **Value** |
| \*myRedshiftPassword | “Your Password” |
| myRedshiftSecurityGrps | “Your security group role” |
| myRedshiftDbName | “Your Redshift DB Name” |
| myRedshiftUsername | “Your Redshift Username” |
| myRedshiftCreateTableSql (Optional) | Not used in this integration |
| myInsertMode | OVERWRITE\_EXISTING (Not used in this integration) |
| myRedshiftTableName | Not used in this integration |
| myInputS3Loc | “S3 location provided to DitR for uploads” |
| myRedshiftJDBCConnectStr | “Your JDBC connection string to your redshift DB” |
| myRedshiftCopyOpts (Optional) | Not used in this integration |
| myPrimaryKeys (Optional) | Not used in this integration |

1. Normalizing and Distributing data
2. Users Table (ditr\_users):

|  |  |  |  |
| --- | --- | --- | --- |
| **PK** | **Name** | **Data Type** | **Description** |
|  | Id | varchar(150) | DitR unique identifier |
| X | machineid | varchar(150) | Unique ID set on a client browser that persists until cleared |
| X | user | varchar(2000) | Unique ID set with a set user method from site owner |
|  | createdat | timestamp | DitR server timestamp of event |
|  | updatedat | timestamp | DitR server timestamp of update |
|  | siteid | Varchar(150) | Unique identifier provided by DitR per website integration |

AWS Datapipline Configuration:

Activities:

|  |  |
| --- | --- |
| **Field** | **Value** |
| Name | Clear\_Users |
| Type | SqlActivity |
| Database | RedshiftCluster |
| Runs On | Ec2Instance |
| Script | Delete from ditr\_users where id in (select du.id from ditr\_users du inner join events\_staging es on du.machineid= es.machineid and du.”user”=es.”user”) |
| Depends On | JSON\_to\_Staging |

|  |  |
| --- | --- |
| **Field** | **Value** |
| Name | Load\_Users |
| Type | SqlActivity |
| Database | RedshiftCluster |
| Runs On | Ec2Instance |
| Script | Insert into ditr\_users (select id, machineid, "user", cast(createdat as timestamp), cast(updatedat as timestamp), siteid from (select id, machineid, "user", cast(createdat as timestamp), cast(updatedat as timestamp), siteid,row\_number() over(partition by machineid, "user" order by cast(createdat as timestamp) desc ) as "rn" from events\_staging where "user" is not null) where rn=1); |
| Depends On | Clear\_Users |

1. Sessions Table (ditr\_sessions):

|  |  |  |  |
| --- | --- | --- | --- |
| **PK** | **Name** | **Data Type** | **Description** |
| X | id | varchar(150) | DitR unique identifier |
|  | sessionid | varchar(150) | DitR session identifier |
|  | browser | varchar(150) | Client browser |
|  | sourceutm | varchar(2000) | Session source based on UTM parameters |
|  | mediumutm | varchar(2000) | Session medium based on UTM parameters |
|  | campaignutm | varchar(2000) | Session campaign based on UTM parameters |
|  | termutm | varchar(2000) | Session term based on UTM parameters |
|  | contentutm | varchar(2000) | Session content based on UTM parameters |
|  | operatingsystem | varchar(150) | Operating system of client machine |
|  | referrer | varchar(2000) | Referrer of session |
|  | machineid | varchar(150) | Unique ID set on a client browser that persists until cleared |
|  | ip | varchar(150) | IP address of client machine |
|  | country | varchar(150) | Country of client based on IP location |
|  | region | varchar(150) | Region of client based on IP location |
|  | city | varchar(150) | City of client based on IP location |
|  | zip | varchar(150) | Zip code of client based on IP location |
|  | timezone | varchar(150) | Time zone of client based on IP location |
|  | latitude | varchar(150) | Latitude of client based on IP location |
|  | longitude | varchar(150) | Longitude of client based on IP location |
|  | createdat | timestamp | DitR server timestamp of event |
|  | updatedat | timestamp | DitR server timestamp of update |
|  | time | timestamp | Client time of event |
|  | siteid | varchar(150) | Unique identifier provided by DitR per website integration |
|  | device | varchar(150) | Device type of client (Mobile, Tablet, Desktop) |

AWS Datapipline Configuration:

Activities:

|  |  |
| --- | --- |
| **Field** | **Value** |
| Name | Load\_Sessions |
| Type | SqlActivity |
| Database | RedshiftCluster |
| Runs On | Ec2Instance |
| Script | Insert into ditr\_sessions (select id, sessionid, browser, sourceutm, mediumutm, campaignutm, termutm, contentutm, operatingsystem, referrer, machineid, ip, country, region, city, zip, timezone, latitude, longitude, cast(createdat as timestamp), cast(updatedat as timestamp), cast("time" as timestamp), siteid, device from(  select id, sessionid, browser, sourceutm, mediumutm, campaignutm, termutm, contentutm, operatingsystem, referrer, machineid, ip, country, region, city, zip, timezone, latitude, longitude, cast(createdat as timestamp), cast(updatedat as timestamp), cast("time" as timestamp), siteid, device, row\_number() over(partition by sessionid order by cast(createdat as timestamp)) as "rn" from events\_staging) where rn=1); |
| Depends On | JSON\_to\_Staging |

1. Pages Table (ditr\_pages):

|  |  |  |  |
| --- | --- | --- | --- |
| **PK** | **Name** | **Data Type** | **Description** |
| X | id | varchar(150) | DitR unique identifier |
|  | sessionid | varchar(150) | DitR session identifier |
|  | url | varchar(65535) | Full URL of webpage |
|  | urlquery | varchar(65535) | Query string of URL |
|  | eventnumber | integer | Sequential order of events in session |
|  | machineid | varchar(150) | Unique ID set on a client browser that persists until cleared |
|  | createdat | timestamp | DitR server timestamp of event |
|  | updatedat | timestamp | DitR server timestamp of update |
|  | time | timestamp | Client time of event |
|  | siteid | Varchar(150) | Unique identifier provided by DitR per website integration |
|  | pagelabel | varchar(150) | Page label set in the DitR tracking by user website |
|  | pagetype | varchar(150) | Page type set in the DitR tracking by user website |

AWS Datapipline Configuration:

Activities:

|  |  |
| --- | --- |
| **Field** | **Value** |
| Name | Load\_Pages |
| Type | SqlActivity |
| Database | RedshiftCluster |
| Runs On | Ec2Instance |
| Script | Insert into ditr\_pages (select id, sessionid, url, urlquery, eventnumber, machineid, cast(createdat as timestamp), cast(updatedat as timestamp), cast(“time” as timestamp), siteid, pagelabel, pagetype from events\_staging where type='pageload') |
| Depends On | JSON\_to\_Staging |

1. Events Table (ditr\_events):

|  |  |  |  |
| --- | --- | --- | --- |
| **PK** | **Name** | **Data Type** | **Description** |
| X | id | varchar(150) | DitR unique identifier |
|  | sessionid | varchar(150) | DitR session identifier |
|  | url | varchar(65535) | Full URL of webpage |
|  | eventnumber | integer | Sequential order of events in session |
|  | type | varchar(150) | click, blur, scroll, pageload |
|  | target | varchar(65535) | Target of an element that is the focus of a click |
|  | screenresolution | varchar(150) | Screen resolution of client machine |
|  | scrollpercent | float | Percent scroll depth on browser page |
|  | windowheight | Integer | Height of browser window |
|  | windowwidth | integer | Width of browser window |
|  | machineid | varchar(150) | Unique ID set on a client browser that persists until cleared |
|  | href | varchar(2000) | Href of a registered click when present |
|  | element | varchar(65535) | Element that is the focus of a click |
|  | xcoordinate | float | X coordinate of click |
|  | ycoordinate | float | Y coordinate of click |
|  | createdat | timestamp | DitR server timestamp of event |
|  | updatedat | timestamp | DitR server timestamp of update |
|  | time | timestamp | Client time of event |
|  | siteid | Varchar(150) | Unique identifier provided by DitR per website integration |

AWS Datapipline Configuration:

Activities:

|  |  |
| --- | --- |
| **Field** | **Value** |
| Name | Load\_Events |
| Type | SqlActivity |
| Database | RedshiftCluster |
| Runs On | Ec2Instance |
| Script | Insert into ditr\_events (select id, sessionid, url, eventnumber, type, target, screenresolution, scrollpercent, windowheight, windowwidth, machineid, href, element, xcoordinate, ycoordinate, cast(createdat as timestamp), cast(updatedat as timestamp), cast("time" as timestamp), siteid from events\_staging where type<>'pageload') ; |
| Depends On | JSON\_to\_Staging |