Write Azure SQL to process Event data into daily usage data

Outline:

We have event data coming in as users do work in our software. This needs to be collated into analytical data to provide the user with information as to their usage.

Users can create and run “Elements” and “Boards” in our software. A Board is a series of Elements opened within a Layout on the screen. E.g.: a requirements doc, a project IDE and a bug tracker are all elements. A board would open all three of these across the screen using the layout you have specified opening them all at once. I can share these with other users to use.

We want to be able to provide users on which elements and boards they are using, how log they are using and if they are popular with other users. There will be a report element that will build on top of the new data to turn it into something more human readable.

The challenge here is to collate the raw event data into a data set which is more suited to fast reporting.

The script will be run daily; the intention is to run the processing daily to review the last 24 hours of event logs, but if earlier event logs have not been processed yet then they will be batched into daily chunks and processed together.

The aim is to provide the users with details of their daily activity and trends over time.

Assumptions:

I imagine this would be written in Azure SQL that can be executed daily to process the previous day’s event data. Please structure the code well and provide extensive commenting to make it easier to maintain.

This will be version 1.0 and process only a few items. The intention is to expand with time. There may be follow up work depending upon the success of this initial work.

NB: We never actually include any User information beyond their ID, it will never be accessible from this data. Users will always just be referred to using their UUID. This is intended to provide a level of user privacy. A user will be able to get a report on their data by providing their UUID. Everyone else will be anonymised in any reports built on the data.

Error handling: the source data may never be perfect. If there is critical missing data [i.e. a required field being NULL] then the update will be ignored and execution will carry on. It would be good if these are logged as an error in a new table (you can specify) for review afterwards.

The Data:

The following SQL connects to the existing data sources and establishes a series of new tables into which the processed data will be inserted.

This SQL connects to an external DB (read-only) containing all the source data and creates the tables to hold the processed data set. I will provide more explanation afterwards.

```
/* Verified Works */

/* ON MASTER DB */
CREATE LOGIN analyticsdb WITH PASSWORD = 'analyticsdbPW'

/* On PRODUCT SERVER */
CREATE USER analyticsdb FOR LOGIN analyticsdb
GRANT SELECT ON [dbo].[AnalyticsEventTypes] TO analyticsdb
GRANT SELECT ON [dbo].[AnalyticsEvents] TO analyticsdb
GRANT SELECT ON [dbo].[Boards] TO analyticsdb
GRANT SELECT ON [dbo].[BoardSharing] TO analyticsdb
GRANT SELECT ON [dbo].[Elements] TO analyticsdb
GRANT SELECT ON [dbo].[ElementSharing] TO analyticsdb



/* On PRODUCT ANALYTICS SERVER */
/* CREATE DATABASE sqldb-PRODUCT-analytics */
BEGIN -- Establish sqldb-PRODUCT-analytics INTERNAL tables
CREATE TABLE [ProcessingHistory] (
[NumericId] int IDENTITY(1,1) PRIMARY KEY,
[LastAnalyticsEventsNumericId] int NOT NULL,
[LastAnalyticsEventsProcessDate] date NOT NULL,
[EntryTimeStamp] datetimeoffset NOT NULL,
[LogsProcessed] int
)

CREATE TABLE [UserSessions] (
[NumericId] int IDENTITY(1,1) PRIMARY KEY,
[UserId] uniqueidentifier NOT NULL,
[DataDate] date NOT NULL,
[SessionId] uniqueidentifier NOT NULL,
[SessionLength] int
)

CREATE TABLE [BoardDetails] (
[Id] uniqueidentifier PRIMARY KEY,
[Name] nvarchar(100) NOT NULL,
[Created] datetimeoffset NOT NULL,
[CreatedDate] date NOT NULL,
[OwnerUserId] uniqueidentifier NOT NULL,
[SharedCount] int,
[LastRan] datetimeoffset,
[LastEdited] datetimeoffset,
[LastEditedDate] date NOT NULL,
[NumberofElements] int,
[Archived] bit NOT NULL
);

CREATE TABLE [BoardRuns] (
[Id] int PRIMARY KEY,
CONSTRAINT [FK_BoardDetails_Id] FOREIGN KEY (BoardId) REFERENCES [BoardDetails](Id),
[BoardId] uniqueidentifier NOT NULL,
[UserId] uniqueidentifier NOT NULL,
[DataDate] date NOT NULL,
[RunTime] int
)

CREATE TABLE [ElementDetails] (
```

```
[Id] uniqueidentifier PRIMARY KEY,
[Name] nvarchar(100) NOT NULL,
[Created] datetimeoffset NOT NULL,
[CreatedDate] date NOT NULL,
[OwnerUserId] uniqueidentifier NOT NULL,
[SharedCount] int,
[LastRan] datetimeoffset,
[LastEdited] datetimeoffset,
[LastEditedDate] date NOT NULL,
[Archived] bit NOT NULL
)

CREATE TABLE [ElementRuns] (
[Id] int PRIMARY KEY,
[ElementId] uniqueidentifier NOT NULL,
CONSTRAINT [FK_ElementDetails_Id] FOREIGN KEY (ElementId) REFERENCES [ElementDetails](Id),
[UserId] uniqueidentifier NOT NULL,
[DataDate] date NOT NULL,
[RunTime] int
)
END

BEGIN
CREATE MASTER KEY ENCRYPTION BY PASSWORD = 'encryptionPW';
CREATE DATABASE SCOPED CREDENTIAL QueryCredential
  WITH IDENTITY = 'analyticsdb', SECRET = 'analyticsdbPW'

CREATE EXTERNAL DATA SOURCE PRODUCTDB
 WITH
 ( TYPE = RDBMS,
   LOCATION='server.database.windows.net',
   DATABASE_NAME = 'sqldb-PRODUCT',
   CREDENTIAL = QueryCredential
 );

-- Set tables
CREATE EXTERNAL TABLE [AnalyticsEventTypes] (
    [NumericId] int NOT NULL,
    [Id] uniqueidentifier NOT NULL,
    [Name] nvarchar(100) NOT NULL
) WITH (DATA_SOURCE = PRODUCTDB)

CREATE EXTERNAL TABLE [AnalyticsEvents] (
    [NumericId] int NOT NULL,
    [Id] uniqueidentifier NOT NULL,
    [TypeId] uniqueidentifier NOT NULL,
    [SessionId] uniqueidentifier NOT NULL,
    [CorrelationId] uniqueidentifier NULL,
    [UserId] uniqueidentifier NULL,
    [Date] datetimeoffset NOT NULL,
    [Properties] nvarchar(MAX) NOT NULL,
    [ClientInfo] nvarchar(MAX) NOT NULL
) WITH (DATA_SOURCE = PRODUCTDB)

CREATE EXTERNAL TABLE [Boards] (
    [NumericId] int NOT NULL,
    [Id] uniqueidentifier NOT NULL,
    [Name] nvarchar(100) NOT NULL,
    [Description] nvarchar(500) NOT NULL,
    [CreatedBy] uniqueidentifier NOT NULL,
    [CreatedDate] datetimeoffset NOT NULL,
    [ModifiedBy] uniqueidentifier NULL,
    [ModifiedDate] datetimeoffset NULL,
    [ArchivedBy] uniqueidentifier NULL,
    [ArchivedDate] datetimeoffset NULL,
    [Archived] bit NOT NULL
) WITH (DATA_SOURCE = PRODUCTDB)

CREATE EXTERNAL TABLE [BoardSharing] (
    [NumericId] int NOT NULL,
    [Id] uniqueidentifier NOT NULL,
```

```
    [SharedBy] uniqueidentifier NOT NULL,
    [SharedTo] uniqueidentifier NOT NULL,
    [SharedDate] datetimeoffset NOT NULL,
    [Level] nvarchar(50) NOT NULL,
    [Seen] bit NOT NULL,
    [Assigned] bit NOT NULL,
    [BoardId] uniqueidentifier NOT NULL
) WITH (DATA_SOURCE = PRODUCTDB)

CREATE EXTERNAL TABLE [Elements] (
    [NumericId] int NOT NULL,
    [Id] uniqueidentifier NOT NULL,
    [Name] nvarchar(500) NOT NULL,
    [CreatedBy] uniqueidentifier NOT NULL,
    [CreatedDate] datetimeoffset NOT NULL,
    [ModifiedBy] uniqueidentifier NULL,
    [ModifiedDate] datetimeoffset NULL,
    [ArchivedBy] uniqueidentifier NULL,
    [ArchivedDate] datetimeoffset NULL,
    [Archived] bit NOT NULL,
    [TemplateVariantId] uniqueidentifier NOT NULL,
) WITH (DATA_SOURCE = PRODUCTDB)

CREATE EXTERNAL TABLE [ElementSharing] (
    [NumericId] int  NOT NULL,
    [Id] uniqueidentifier NOT NULL,
    [SharedBy] uniqueidentifier NOT NULL,
    [SharedDate] datetimeoffset NOT NULL,
    [SharedTo] uniqueidentifier NOT NULL,
    [Level] nvarchar(50) NOT NULL,
    [Seen] bit NOT NULL,
    [ElementId] uniqueidentifier NOT NULL
) WITH (DATA_SOURCE = PRODUCTDB)
END
```

I can summarise the source data tables text and provide examples and more information

Raw Input Data Sets – Existing Tables in DB (Read-only)

AnalyticsEventTypes: This table stores information about the types of analytics events, with a:

- NumericId as the primary key – not need
- Id (uniqueidentifier) – which corresponds to the TypeID in the AnalyticsEvents table
- Name (nvarchar(100)) – human readable version of TypeID – helps you understand what is happening

The full data in this table:

NumericId	Id	Name
1	1713A5AE-2F98-49B0-B3C8-3875213E96DD	ElementCreated
2	CF6119BB-3ABC-4BA4-96E7-B9DEDEA6A405	ElementUpdated
3	F09AC448-66E9-4FEF-A0F9-066406F2A094	ElementDeleted
4	D4F1B876-190F-437D-9B4A-72A2E58BA85B	ElementShared
5	2B5C43C1-07FF-439F-952A-703D3DAD1177	ElementStarted
6	35525E1E-3210-4C32-9D77-6458B2ED2F66	ElementStopped
7	EE1F923E-2E8C-457A-8FE5-F6D4EFA6D085	BoardCreated
8	52C953F5-01D1-4D92-8913-E14342FE9A1A	BoardUpdated
9	06F9500E-52B6-4119-A75C-9B12136F9A75	BoardDeleted
10	D50F0C47-28B3-4106-8801-99564BE8451E	BoardShared
11	A55472D9-5246-49C1-92F1-DD047C2492F5	BoardStarted
12	C164B223-0FBB-4B8C-B2EF-A679D9F38988	BoardStopped
13	4DE82C5A-1475-4FD0-A455-AC8DD8A44B0E	EnvironmentCreated
14	336ECCED-5D91-4A79-8D42-25BE88B9C7A0	EnvironmentUpdated
15	E228FD9C-1ED5-4D44-9B82-094220A83F4A	EnvironmentDeleted
16	DACB8E1B-F5AF-4F75-A727-8243B2851291	EnvironmentUsed
17	DF10B43E-7EF8-4A46-87D9-06EC291BFF97	LayoutCreated
18	F0DABF54-50D4-4FE7-BB58-FD575F8E7F81	LayoutUpdated
19	FA8B7F3A-0872-460A-86F2-5521776FEEB4	LayoutDeleted
20	708AC6CA-509C-4945-84FF-3BAB0F21CF6B	LayoutUsed
21	084F0CB9-0BF1-4F83-970A-2628402D3E9F	FolderCreated
22	45924F82-D00B-442B-AC88-F98D3075FE09	FolderUpdated
23	9254F255-2D10-41C8-A999-67D7A9B66FA6	FolderDeleted
24	DC7FC2E6-CEEF-41CC-BC84-6656C64187BB	TagCreated
25	757301AE-2FCD-4E18-8315-F3045891A642	TagDeleted
26	AC87938F-14E6-4EED-8022-495949C2664B	TemplateCreated
27	04CE32A0-98D4-4651-BE5E-A39B4F38FEE9	TemplateUpdated
28	DE15B52E-7AFB-4778-BE30-12A27F01ADC0	TemplateDeleted
29	4072F0A2-2587-4E69-BFC3-6EAF10F6E482	UserGroupCreated
30	838E1979-BC53-41D1-BBFE-787F459D5F64	UserGroupUpdated
31	98851682-A957-43A0-B78A-E4D0492E5161	UserGroupDeleted
32	4440DBA8-B287-4038-8085-82BD2B0E10AC	RoleCreated
33	68850F3E-223D-47E1-A5BC-359D86E78FA4	RoleUpdated
34	DA5E7AD0-D2FA-42CC-AE39-75A59D7ACF48	RoleDeleted
35	B4DB8BAB-D5C0-4F06-8A32-A2BF796C132B	UserCreated
36	0B85AEC4-24FF-4222-BC5D-DD60AB5E6974	UserUpdated
37	C40ABB13-2957-4FF2-993E-71441F50C8EC	UserDeleted

We are only interested in the initial 12 event types for now – **For now grey entries can be ignored**

AnalyticsEvents: This table stores the actual analytics events. It contains

- NumericId as the primary key – just an auto-increment, oldest data will be 1
- Id (uniqueidentifier) – UUID of the event – not especially useful
- TypeId (uniqueidentifier) as a foreign key referring to the AnalyticsEventTypes table
- SessionId (uniqueidentifier)
- CorrelationId (uniqueidentifier) – which can act as a foreign key to another table (see below)
- UserId (uniqueidentifier)
- Date (datetimeoffset)
- Properties (nvarchar(MAX)): JSON field - details from the Event – varies according to type
- ClientInfo (nvarchar(MAX)): JSON field – details of the client - consistent
- Version – of the current version – so we can cope with changes to event data in the future

For now grey entries can be ignored as we will not be needing them, just yet.

Here is some example Analytics Event data:

NumericId	Id	TypeId	SessionId	CorrelationId	UserId	Date	Properties	ClientInfo	Version
703	AA7D4297-1966-4A78-BDE5-A30CA7C4C70B	A55472D9-5246-49C1-92F1-DD047C2492F5	7B529A7A-ED6F-4767-857C-6111A8DDE2BE	6F47CF6F-9A43-49E0-B97A-24D8AB35E346	42B8BC6A-3068-4EC2-8015-626AF6C61E2A	2022-09-14 07:11:48.836456 7+00:00	{"elementIds":["16a94937-1b26-4873-a305-3c14b8655536","9d916c0b-96a2-4bf4-b7b6-b5907547d073","90691940-4c8a-4bd2-9df6-2cbe8e678162","4ddbf4ac-e9f8-42cc-b2e5-457dd5436bb4","8881172d-eab0-41ba-98f2-b16787bcbfc3","64bbff31-df2b-4523-aed3-1f6b4688fa44"],"layoutId":"11fa8cd0-9088-4cc1-9810-8c75d464d56d"}	{"ipAddress":"89.10.176.159","userAgent":"Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/105.0.0.0 Safari/537.36 Edg/105.0.1343.33","operatingSystem":"Windows 10","device":"Other","browser":"Edge 105.0.1343"}	1.7.1+473b5bcd
704	CE611E70-2BFD-442D-B1BA-3C4B1154C5A6	A55472D9-5246-49C1-92F1-DD047C2492F5	51C5C11C-B9F3-45DF-AB02-3EFFCFEE68ED	7AD302CA-EC29-46A1-B75B-DE7C022142AB	882FE724-E88E-4525-B503-7FE5CB44310E	2022-09-14 07:47:23.0778579 +00:00	{"elementIds":["9b00f6ef-fa33-433f-815f-04bd84bf35c5","0add8bed-5bc2-4569-b093-b457a7d9d141","da1bce33-67c6-4fd9-8de6-fef08581394c"],"layoutId":"dd34ea47-98a0-41c3-bf11-4e8dd02b4a42"}	{"ipAddress":"88.91.17.240","userAgent":"Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/105.0.0.0 Safari/537.36 Edg/105.0.1343.27","operatingSystem":"Windows 10","device":"Other","browser":"Edge 105.0.1343"}	1.7.1+473b5bcd
705	09496191-7187-4C6E-87B5-319F241090C5	2B5C43C1-07FF-439F-952A-703D3DAD1177	E3E9919D-14BB-43C8-A14A-B0E041D1251C	27D4DFB1-AC3F-43C5-9344-90B54C49A122	083DCFF7-90FE-44A9-A2C7-AC1F09D0C442	2022-09-14 07:48:45.5013217 +00:00	{"displayId":"cfddc933-b9ba-4f17-a2e2-cd28a7ae626d","searchTime":1231,"windowFound":true,"error":null}	{"ipAddress":"62.24.36.77","userAgent":"Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/105.0.0.0 Safari/537.36 Edg/105.0.1343.33","operatingSystem":"Windows 10","device":"Other","browser":"Edge 105.0.1343"}	1.7.1+473b5bcd
706	ED69D825-88BE-4108-9C06-87E12CF8C263	C164B223-0FBB-4B8C-B2EF-A679D9F38988	51C5C11C-B9F3-45DF-AB02-3EFFCFEE68ED	7AD302CA-EC29-46A1-B75B-DE7C022142AB	882FE724-E88E-4525-B503-7FE5CB44310E	2022-09-14 07:54:38.3693951 +00:00	{"layoutId":"dd34ea47-98a0-41c3-bf11-4e8dd02b4a42"}	{"ipAddress":"88.91.17.240","userAgent":"Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/105.0.0.0 Safari/537.36 Edg/105.0.1343.27","operatingSystem":"Windows 10","device":"Other","browser":"Edge 105.0.1343"}	1.7.1+473b5bcd
707	3449D545-B4A4-452F-A547-29C774A92E9F	A55472D9-5246-49C1-92F1-DD047C2492F5	51C5C11C-B9F3-45DF-AB02-3EFFCFEE68ED	7AD302CA-EC29-46A1-B75B-DE7C022142AB	882FE724-E88E-4525-B503-7FE5CB44310E	2022-09-14 07:54:59.9657965 +00:00	{"elementIds":["9b00f6ef-fa33-433f-815f-04bd84bf35c5","0add8bed-5bc2-4569-b093-b457a7d9d141","da1bce33-67c6-4fd9-8de6-fef08581394c"],"layoutId":"dd34ea47-98a0-41c3-bf11-4e8dd02b4a42"}	{"ipAddress":"88.91.17.240","userAgent":"Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/105.0.0.0 Safari/537.36 Edg/105.0.1343.27","operatingSystem":"Windows 10","device":"Other","b	1.7.1+473b5bcd

								rowser":"Edge 105.0.1343"}	
708	3CEFB40E-6DEE-42CB-8F29-D4C2109078A0	C164B223-0FBB-4B8C-B2EF-A679D9F38988	51C5C11C-B9F3-45DF-AB02-3EFFCFEE68ED	7AD302CA-EC29-46A1-B75B-DE7C022142AB	882FE724-E88E-4525-B503-7FE5CB44310E	2022-09-14 07:55:12.6423339 +00:00	{"layoutId":"dd34ea47-98a0-41c3-bf11-4e8dd02b4a42"}	{"ipAddress":"88.91.17.240","userAgent":"Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/105.0.0.0 Safari/537.36 Edg/105.0.1343.27","operatingSystem":"Windows 10","device":"Other","browser":"Edge 105.0.1343"}	1.7.1+473b5bcd
709	B9A1C4E7-D4CE-4C64-ACD8-20174B946E1D	2B5C43C1-07FF-439F-952A-703D3DAD1177	E3E9919D-14BB-43C8-A14A-B0E041D1251C	F5FF02D7-8478-4B40-8349-DAD5F6752CA6	083DCFF7-90FE-44A9-A2C7-AC1F09D0C442	2022-09-14 07:57:22.2773725 +00:00	{"displayId":"a622b6df-a0cf-492d-a648-b6852db54433","searchTime":1770,"windowFound":true,"error":null}	{"ipAddress":"62.24.36.77","userAgent":"Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/105.0.0.0 Safari/537.36 Edg/105.0.1343.33","operatingSystem":"Windows 10","device":"Other","browser":"Edge 105.0.1343"}	1.7.1+473b5bcd
710	28D8511B-92AA-48D8-9917-419F3E5ADD10	2B5C43C1-07FF-439F-952A-703D3DAD1177	7BBB2F2A-84A4-4A28-832E-445D861F5804	4524B0C0-BE27-4E4E-9AE5-FD75794416 27	B89A0C80-4228-4B21-8D63-CFA344ECC3CC	2022-09-14 07:59:55.1146697 +00:00	{"displayId":"012f641d-6ae4-4af2-991b-0e5f4cb60184","searchTime":12136,"windowFound":true,"error":null}	{"ipAddress":"85.165.10.30","userAgent":"Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/105.0.0.0 Safari/537.36 Edg/105.0.1343.27","operatingSystem":"Windows 10","device":"Other","browser":"Edge 105.0.1343"}	1.7.1+473b5bcd
711	3EAB9A3F-2358-4BBF-93A7-E4E9797647DE	2B5C43C1-07FF-439F-952A-703D3DAD1177	E3E9919D-14BB-43C8-A14A-B0E041D1251C	DBC7F1CF-27F0-42B5-B673-5C50ABE2157A	083DCFF7-90FE-44A9-A2C7-AC1F09D0C442	2022-09-14 08:00:06.7849650 +00:00	{"displayId":"a622b6df-a0cf-492d-a648-b6852db54433","searchTime":1528,"windowFound":true,"error":null}	{"ipAddress":"62.24.36.77","userAgent":"Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/105.0.0.0 Safari/537.36 Edg/105.0.1343.33","operatingSystem":"Windows 10","device":"Other","browser":"Edge 105.0.1343"}	1.7.1+473b5bcd

The **`CorrelationId`** always refers to another entity within the database – at the moment I am limiting that to Element & Board events as mentioned above.

For Board Events the **`CorrelationId`** is the **`Boards.ID`**

- Note the actual elements this board opens are not tracked in the sample data, but they are listed in the event Properties – which you can ignore for now apart from just to count the number of elements opened.
- In the real data set the elements opened in this way will be tracked like every other element. So we can ignore the actual elements and layouts being used as this is just a complicating factor for now as it will self-resolve.

For Element Events the **`CorrelationId`** is the **`Elements.ID`**

I can provide large sample sets of data if required. Just ask.

Boards: This table contains information about the boards, with a:

- NumericId as the primary key
- Id (uniqueidentifier)
- Name (nvarchar(100))
- Description (nvarchar(500))
- CreatedBy (uniqueidentifier) as the user id who created the board
- CreatedDate (datetimeoffset)
- ModifiedBy (uniqueidentifier) as the user id who last modified the board
- ModifiedDate (datetimeoffset)
- ArchivedBy (uniqueidentifier) as the user id who archived the board
- ArchivedDate (datetimeoffset)
- Archived flag (bit) to indicate if the board is archived.

BoardSharing: This table stores information about board sharing between users. It has a:

- NumericId as the primary key
- Id (uniqueidentifier)
- SharedBy (uniqueidentifier) as the user id who shared the board
- SharedTo (uniqueidentifier) as the user id the board was shared with
- SharedDate (datetimeoffset)
- Level (nvarchar(50)) as the access level
- Seen (bit) to indicate if the shared board was seen by the recipient
- Assigned (bit) to indicate if the shared board was assigned to the recipient
- BoardId (uniqueidentifier) as a foreign key referring to the Boards table.

Elements & **Element Sharing** are much the same as Boards.

Some of this data will be needed, some not. It should be obvious.

The New Processed Data Tables & how they are populated.

ProcessingHistory – update daily (or more if processing multiple days)

This table is just used to track the processing completed – so when the SQL scripts are run again they have the context to restart.

The structure of the ProcessingHistory table should be as follows:

- NumericId: An integer value that auto-increments and serves as the primary key for each record in the ProcessingHistory table.
- LastAnalyticsEventsNumericId: An integer representing the NumericId of the last processed event in the AnalyticsEvents table.
- LastAnalyticsEventsProcessDate: The date when the last processing of AnalyticsEvents took place.
- EntryTimeStamp: The date and time (with timezone offset) when the record was added to the ProcessingHistory table.
- LogsProcessed: An integer representing the total number of logs processed in the corresponding batch today.

Every time you process a days' worth of data you add an entry to this table. It lets us keep track of processing and to help debug. If you are processing several days of data at once (due to a backlog) each day will be processed separately and an entry added.

If there is no previous processing history it starts processing from the date of the first event in the AnalyticsEvents table.

Whenever we refer to a day's worth of events – assume a local calendar day; midnight to midnight. We will schedule the SQL to run a few hours after this time (e.g.: 3am)

UserSessions – update daily – new entries added – edge case to edit old entry

This table just tracks the length of each user session. It will be easy to use this table to create reports on the number of sessions a day, average length of sessions, number of active users a day, etc.

- NumericId: An integer value that auto-increments and serves as the primary key for each record in the table.
- UserId: the user that initiated the session
- DataDate: the date the session started on
- SessionID: The session ID being tracked
- SessionLength: the time in seconds between the first and last events of the selected SessionID. Sessions may rarely run over the processing day, so it will be necessary to check the current SessionID is unique and if not update the previous entry in the table with a new SessionLength rather than create a new entry.

BoardDetails – entries only created & updated by particular events

This table largely pulls across the relevant data from the input Boards data. New entries are added as new boards are added (this can be done daily). The data in the table is just updated otherwise.

The BoardRuns table (next) holds the dynamic usage data.

- Id: The Boards.Id - copied from the Boards table at creation time
- Name: Boards.Name - copied from the Boards table at creation time – if board is edited – update this
- Created: The date this entry was created
- CreatedDate: Boards.CreatedDate - Copied from the Boards table at creation time
- OwnerUserId: Boards.CreatedBy - Copied from the Boards table at creation time
- SharedCount: The number of times the BoardId appears in the BoardSharing table – this will need to be updated with any BoardSharing eventTypes
- LastRan: The date the board was last ran – this will need to be updated with when there is a BoardStarted event type
- LastEdited: The userId of the user who last updated or deleted the board– this will need to be updated with when there is a BoardUpdated or BoardDeleted event type
- LastEditedDate: The date the board was last updated or deleted – this will need to be updated with when there is a BoardUpdated or BoardDeleted event type
- NumberofElements: a quick count of the ElementIds – this can be determined when you get a BoardStarted event by counting the number of ElementIds in the Event Properties.
- Archived: If the board is archived (effectively deleted) - Copied from the Boards.Archived at creation time - this will need to be updated (recopied) with when there is a BoardUpdated or BoardDeleted event type

Remember that for Board related Events the CorrolationId is the BoardId

BoardRuns – update daily – new entries added

A new entry is added to this table after every BoardStarted event type. There *should* be a corresponding BoardStopped event to get all the data, however if this is missing have a runtime of -1 seconds. That means we can still count the board as run, but we can ignore the data for average run-times, etc.

- Id: probably should just be an incrementing Id
- BoardId: The BoardId of the board that was started – from BoardStarted Event
- UserId: The UserId of the user who started the board – from BoardStarted Event
- DataDate: The date when the board was run – from BoardStarted Event
- RunTime: the time in seconds between the BoardStarted Event and the corresponding BoardStopped Event (Should have matching UserId, SessionId & CorrleationId)

There may be a possibility that a BoardStopped event is not recorded for a board if they just start a new board or shut down their PC.

If there are multiple BoardStarted Events for the same UserId/SessionId/CorrleationId you can only associate the BoardStopped event with the last one. Any other Board Started events will have a runtime of -1. This may be updated in the future – we may just assume the next user event acts as a

BoardsStopped event – but we can't make that judgment until we see the real state of the data and how often this occurs. So comment as such in the code and we can revisit later if needed.

ElementDetails & **ElementRuns** are much the same as above. Slight variances an simplifications. Ask if you have any queries.