Documentation

Mavroudo

March 24, 2020

1 Model

Describing all the classes that are represented in the model package

- **AugmentedDetail:** Represents an detail that is augmented by the event name to which it corresponds. Constructor (eventName,key,value). Compare, Equals, toString, hashCode.
- **Completion:** Represents a Completion returned through the quick_stats endpoint. Constructor (step,completions,averageDuration,lastCompletedAt). *The step index of the completed funnel (e.g. in a funnel A-¿B-¿C a completion with step index 1 corresponds to a completion of the A-¿B sub-funnel).* Getters,Setter,toString.
- **Detail:** A Step Detail as provided by the JSON input. Is a Builder has 3 fields: key, operator, value. toString returns a json file.
- **DetailedCompletion:** Represents a single Detailed Completion returned through the export_completions endpoint. Constructor (step, completed_at, duration, session_id, app_id, device_id, user_id). The duration of the funnel, completed_at = the timestamp of the last event in the completion.
- **Event** Represents a funnel Event. 3 different constructors. Fields: *name* (Funnel event name (prefixed by AppID and LogtypeID, i.e. appID_logtypeID_eventName)) and *details* (TreeSet< *AugmentedDetail* >). Compare, matches, toString, equals, hashcode, getPrefixedAppID.
- **EventPair:** 2 events as string_name. Equals, hashcode, equals, toString.
- **EventPairFrequency:** Represents an EventPairFrequency returned by probing the "count" index tables. Constructor (EventPair, sum_duration(sum of all durations of this event pair), comp_count(sum of all completions of this event pair)). Compare (which is not consistent with equals).
- **Funnel:** A Funnel body as provided by the JSON input (i.e. not wrapped by a funnel tag. Contains a Builder. fields: *steps List* < *Step* > and *maxDuration*. A toString that returns a json of this object.

- **FunnelWrapper:** A Funnel Wrapper as provided by the JSON input (i.e. the funnel tag that wraps a funnel body). Contains Builder. has only one field: Funnel.
- **LifeTime:** Represents the lifetime of a DetailedCompletion. Constructor (start_date, end_date, duration, appID).
- **Name:** A Step Name as provided by the JSON input. Builder, fields: *logName*, *applicationID*, *logType*.
- **Proposition:** Represents a Proposition corresponding to a continuation provided by the explore/* endpoint. 3 Constructors. Fields: *applicationID*, *logType*, *logName*, *completions* , *averageDuration*, *lastCompleted*. toString, compareTo.
- **QueryPair:** A pair of events (along with their details) corresponding to a query pair (e.g. a 3-sized funnel has 3 query pairs). Fields: *first_ev*, *second_ev*. hashCode, equals, toString.
- **QueryStatsResponse:** Represents the output of an quick_stats endpoint query. Fields: *completions List < Completion >*. equals, toString, hashCode.
- **Sequence:** Events are stored in an ArrayList in the order they appear in the funnel. Fields: *seq List < Event >*. Methods:
  - *appendToSequence(Event)*
  - *prependToSequence(Event)*:add it to the first position
  - *removeLastEvent*
  - *insert(event,position)*
  - *getList*
  - *getFirstEvent*
  - *getLastEvent*
  - *getQueryTuples()*: return a *List < QueryPair >* (a list of all query pairs for the funnel. For example, a funnel A-B-C has the following query pairs: (A,B), (B,C), (A,C)),
  - *getEvent(position)*
  - *getSize()*
  - *fitsTimestamps(List < TimestampedEvent >, maxDuration)*, Check if this sequence exists in a (superset) sequence of events. Furthermore, the sequence must exist within a time limit. Return an array of two values [timestamp of the first event occurrence, timestamp of the last event occurrence] in the list of timestamped events. If this sequence is not contained in the greater list, [-1,-1] is returned
  - *getUniqueEvents*. Return a Set¡Event¿

- *getSecondLastEventDelimited(String delim).* Get the second to last event of the sequence concatenated with a delimiter (Currently not used).

- **Status:** Represents the output of an @code status endpoint query. Fields: $List < String > user$, $List < String > device$

- **Step:** A Step as provided by the JSON input. Fields: *names* $List < Name >$, *details* $List < Detail >$, which are matched from the json query. Created with Builder.

- **TimestampedEvent:** An event annotated with a ink java.util.Date timestamp. Fields: *timestamp*, *event*. compare, hashCode, equals, toSting.

2 Query

All the classes that are contained in the query package.

- **SequenceQueryHandler:** Class designed to handle the low-level needs of a query evaluation. Each time a query fetches data from the index (not count) tables the incoming rows are handled by this class first. Each row is handled by extracting info about the entity (device or user) ID that achieved the completion and then handling the completion itself in one of two different ways (i.e. as an event pair or as a detail triplet).
  Fields:

  - protected Cluster
  - protected Session
  - protected KeyspaceMetadata
  - protected final string Delimiter
  - $ConcurrentHashMap < String, List < TimestampedEvent >$ allEventsPerSession.

  Methods:

  - public **stripAppIdAndLogtype**
  - public **getYearMonths**. return all months between 2 provided dates + periods per/ 10 days.
  - protected **handleSessionRow**(Sequence, first, second, startDate, endDate,$List < String >$ candSessioons, putInMap). Running through the information per Session and handle each row differently, depending on the data that contains.

- private **handleEventPairs** (dateFormat, sessionID, string[] times, first, second, startDate, endDate). times contains all the different times for this pair of events in this session id. If session id is contained in the allEventsPerSession, then TimestampedEvents will be added, if the timestamps are between limits.
- private **handleDetailPairs**, same as before only now contains a event_key_value.

Classes:

- **CandSessionsCallback** implements FutureCallback< $ResultSet$ >. oveerides onSuccess and onFailure based on if the query in cassandra was successfull or not.

- **SequenceQueryEvaluator** extends **SequenceQueryHandler:** The class responsible for accurately evaluating a query. Each query is evaluated on a per month basis and then all monthly results are aggregated in a single candidate list. All the initial candidates are then tested and any false positives are discarded. **Fields**: No new field, same as the superclass. **Methods**:

  - public Set< $String$ > **evaluateQuery**(startDate, endDate, Sequence, Map< $Integer, List < AugmentedDetail >>$, isUsersQuery). One table per month in Cassandra (dvc_idx_year_month for not userQuery else usr instead of dvc). If table exists then we take the candidates from this month by using **evaluateQueryOnMonth**.
  - private Set< $String$ > **evaluateQueryOnMonth**(tableName, $List <$ $QueryPair >$, Sequence, startDate, endDate, Map< $Integer, List <$ $AugmentedDetail >>$, isUsersQuery). queryPairs was calculated based on the sequence in the previous. Query to Cassandra the first query pair, in the given table, the 2 fields and the third is null. Then continues for the other pairs in a multi threading, suing the **CandSessionsCallBack** from the superclass. Closes all the threads and returns the month candidates.
  - public **findTruePositivesAndLifetime(Sequence, candidates, maxDuration)**. For each candidate we check for two requirements: 1) The events of that candidate occur in the correct order, 2) They occur within the timeframe of the "maxDuration" variable. For every candidate, gets the candidates from the allEventsPerSession. Then using **fitsTimestamps** from Sequence, to find if a pair is contained in a superset. Returns a Map of $< String, LifeTime >$, where string is the candidate pair and lifetime is representation of a DetatiledCompletion.
  - getDeviceIds(startDate, endDate, userIDS), querying Cassandra for every month, every userId and returns the device id.

  - getUserIds(startDate, endDate, deviceIDS), querying Cassandra for every month, every deviceID and returns the user id.

- **SequenceQueryExplorer** extends **SequenceQueryEvaluator:**