Overview

In this challenge you will be asked to take some unstructured data, ingest it into a database, and run some transformations so that it is meaningfully accessible for users.

In particular you will be working with a sample of usage data that could have been generated by mobile devices in Kenyan pre-primary schools. This usage data is represented in the form of events, which are organized in *event logs* (→ *event sourcing*).

1 Tasks

Use tools of your choice to…

(a) Ingest the provided JSON events into the database. The events are described in section 2.

(b) Test the data for any values that are out of range or otherwise invalid.

(c) Write and run the ELT transformations described in section 3.

(d) Unit-test the transformations[1].

(e) Orchestrate tests and transformations.

(f) Generate some basic descriptive statistics about your final dataset (number of schools, devices, sessions, etc.).

(g) Prepare to walk us through your system. What do you like? What would you improve if you had more time? What other tools or services would you bring in?

Tools that are important in our current setup:
dbt (core) with dbt-expectations and dbt-unit-testing, GitHub Actions

You may not be able to finish the entire task in the allotted time — don't worry, that's okay. Just explain what you had planned in the follow-up interview!

If you have any questions or get stuck at any point, please don't hesitate to ask us!

[1] Testing recursive CTEs is currently not possible with Redshift and dbt-unit-testing — you can omit tests for models where you run into this problem.

2 Events

There are three user behaviors that are captured by the events described below:

- the naming of schools
- the assignment of devices to schools
- students playing educational content on the device (referred to as a session)

Here are the event types you can expect to see.

All event types have two fields in common:

- `time` is the creation time of the event. Here you can assume that any clocks are in sync, so `time` can be used to order the events.
- `logId` is a UUID that identifies the log the event belongs to.

SetLogTypeEvent

```
data class SetLogTypeEvent(
    val time: Instant,
    val logId: UUID,
    val logType: String
)
```

Every log needs to have a `SetLogTypeEvent` on it. Here, it can have the values »School« or »Device«.

SetNameEvent

```
data class SetNameEvent(
    val time: Instant,
    val logId: UUID,
    val name: String
)
```

`SetNameEvent` is used for setting (and updating) a school's (or generally any other entity's) name. Here, it should only appear on logs of type »School«.

AssignToSchoolEvent

```
data class AssignToSchoolEvent(
    val time: Instant,
    val logId: UUID,
    val schoolId: UUID
)
```

`AssignToSchoolEvent` assigns a device to a school, effective from the event's timestamp. Hence, it should only appear on »Device« type logs. The value of the `schoolId` field is the log id of the school the device is assigned to. A device can only be assigned to one school at a time.

StartSessionEvent and EndSessionEvent

```
data class StartSessionEvent(
    val time: Instant,
    val logId: UUID,
    val sessionId: UUID
)
```

```
data class EndSessionEvent(
    val time: Instant,
    val logId: UUID,
    val sessionId: UUID
)
```

`StartSessionEvent` and `EndSessionEvent` should appear on »Device« logs and start or end a session, respectively. The session id is chosen by the client and does not refer to a log.

Example

Here are some events for creating and naming two schools, assigning and re-assigning a device to schools, as well as an abandoned and a complete session for the device. The events are in the order of emission, creation times have been omitted for readability.

```
// Initialize school log, set and change the school's name
SetLogTypeEvent(
    ”04d109d0-3a00-4aaa-b3bd-caf95bf74f86”,
    ”School”
)
SetNameEvent(
    ”04d109d0-3a00-4aaa-b3bd-caf95bf74f86”,
    ”My Test Scole”
)
SetNameEvent(
    ”04d109d0-3a00-4aaa-b3bd-caf95bf74f86”,
    ”My Test School”
)
```

```
// Initialize a second school log and set the school's name
SetLogTypeEvent(
    ”10f2c335-9fd3-4e86-9b86-e816cef43a34”,
    ”School”
)
SetNameEvent(
    ”10f2c335-9fd3-4e86-9b86-e816cef43a34”,
    ”Another School Name”
)
```

```
// Initialize a device log, assign the device to a school and
// change that assignment
SetLogTypeEvent(
    ”a6da987a-8a9c-46b0-bbf5-0ef88645b93e”,
    ”Device”
)
AssignToSchoolEvent(
    ”a6da987a-8a9c-46b0-bbf5-0ef88645b93e”,
    ”10f2c335-9fd3-4e86-9b86-e816cef43a34”
)
AssignToSchoolEvent(
    ”a6da987a-8a9c-46b0-bbf5-0ef88645b93e”,
    ”04d109d0-3a00-4aaa-b3bd-caf95bf74f86”
)
```

```
// Start a session (abandoned)
StartSessionEvent(
    "a6da987a-8a9c-46b0-bbf5-0ef88645b93e",
    "5447c740-4cd0-4064-8d94-bfdf9562aeac"
)
```

```
// Start and complete a session
StartSessionEvent(
    "a6da987a-8a9c-46b0-bbf5-0ef88645b93e",
    "62f50bdd-003b-4c5a-8de6-71205af878c3"
)
EndSessionEvent(
    "a6da987a-8a9c-46b0-bbf5-0ef88645b93e",
    "62f50bdd-003b-4c5a-8de6-71205af878c3"
)
```

3 Transformations

Provide the following tables for downstream use:

(a) A list of devices with id and the time when its first session started.

(b) A list of sessions with id, start time, and end time, and the school name the session is associated with.

(c) A history table with a row for each device and day (starting with the day a device first reported a started session), with the following columns:

- `device_id`
- `date`
- `usage_minutes` total session time of device `device_id` on day `date` in minutes (completed sessions only). If no sessions were completed that day `usage_minutes` should be 0. Implement this in a way that supports sessions spanning multiple days! In particular, this should even include cases where a session spans over more than two days.
- `avg_usage_minutes_7d` average daily usage over a seven day window ending at and including `date`

(d) A list of schools with the following columns:

- `school_id`
- `name`
- `device_count` the current number of devices
- `median_session_length` the median session length in minutes of all sessions generated by the schools
- `mean_session_length` the mean session length in minutes of all sessions generated by the schools

4 Provided Information/Data

You should have received the following:

- Access to a GitHub repository (includes access to GitHub actions).
- Credentials for a blank Redshift database.
- An archive of events, serialized to JSON.

Let us know if you need anything else!