Introduction

In the database schema described by this document, elements of a pipeline are broken down into physical entities, processes, calculations, and "compositions". Compositions are entities that are defined by their make-up and are thus immutable--for example, a pool; if you change the ratios of the components in a pool, you have in fact created a *different* pool. Processes and calculations are also immutable--"changing" anything about them creates a new instance of them--while physical entities are mutable in that their contents can change, and they can be discarded/retired/etc. Thsi schema is, undeniably, much more complex than that originally proposed for the 16S process alone, and that is its main drawback. However, it has the following advantages:

1. It accommodates both 16S and shotgun pipelines with minimal duplication of structure or information
2. It provides a generalized model of the type that Austin requested
    - This generalized model should allow a great deal of flexibility for adding new pipelines
3. It tracks process history at the well level rather than the plate level
    - This is necessary to accommodate plate rotations, cherry-picking (not currently performed but anticipated in the future) and interleaving
    - Note that interleaving of four 96-well plates on a single 384-well plate (e.g. first well contains contents from input plate 1, second well contains contents from input plate 2, and so on) already occurs in the shotgun pipeline, and thus must be supported
4. It models the fact that a plate or well can undergo *changes* to its contents
    - This means that treating wells and their contents as interchangeable is inadequate
    - Note that this situation occurs in the shotgun pipeline when a given plate first contains normalized gDNA and then is added to to produce prepped library contents
5. It can represent multiple measurements of the same type made on a single entity
    - This is necessary since sometimes plates are re-quantified if the first quantification seems dodgy
6. It correctly reflects the fact that some processes occur on multiple entities at the same time
    - For example, Greg notes that quantification can be performed on up to three 384-well plates at one time

Besides its complexity, a drawback of this schema is that many processes in the current pipelines ARE in fact conducted at the plate level (e.g., in the 16S pipeline there is a 1:1 relationship between a gDNA plate and a library prep plate), and while this schema can certainly *reflect* this relationship, it cannot *enforce* it with database constraints.

Database Preparation

The following actions would necessary before recording any lab processes in the database.

1. Populate `protocol`

    - primer_template_creation
    - primer_working_plate_creation
    - sample_plating
    - gdna_extraction
    - 16S_library_prep
    - shotgun_library_prep
    - pico_green_quantification
    - qpcr_quantification
    - gdna_normalization
    - manual_pooling
    - automated_pooling

2. Populate `equipment_type`

    - echo
    - mosquito
    - tm 300 8 channel pipette head
    - tm 50 8 channel pipette head
    - etc?

3. Populate `equipment`

    - the echo
    - mosquito 1
    - mosquito 2
    - etc

4. Populate `plate_configuration`

    - 96-well deep-well plate
    - 96-well microtiter plate
    - 384-well microtiter plate
    - etc

5. Populate `composition_type`

    - reagent
    - primer_set
    - sample
    - gdna
    - library_prep
    - pool

6. Populate `reagent_composition_type`

    - master mix specifics
    - water specifics
    - kappa hyper plus kit specifics
    - shotgun stubs specifics

7. Populate `sample_composition_type`

    - experimental sample
    - blank
    - vibrio positive control
    - etc?

8. Populate primer template info
    1. Populate `primer_set`

        - EMP 16S primers
        - I5 shotgun primers
        - I7 shotgun primers

    2. Create `marker_gene_primer_set` record for EMP 16S primers

        - TODO: Is there a need for an analogous subtable for the shotgun primers?

    3. Create `process` record for primer_template_creation

        - will only be one process for this unless/until new primer sets are invented later
        - no need for a specific process subtype here

    4. Create `plate` record for each plate template in each of the primer sets, e.g

        - plate 1 of EMP 16S primer set
        - plate 2 of EMP 16S primer set
        - ...
        - plate 8 of EMP 16S primer set

        - plate 1 of I5 shotgun primer set
        - plate 1 of I7 shotgun primer set
        - hmmm, discarded doesn't make sense for templates

    5. Create `container` and `well` records for each well on each plate template

        - `container`'s `latest_upstream_process_id` will be `process` for primer_template_creation
        - hmmm, remaining volume doesn't make sense for templates

    6. Create `composition` and `primer_set_composition` records for each distinct barcode in each primer set

        - `composition`'s `upstream_process_id` will be `process` for primer_template_creation
        - hmmm, currently total volume is required, but doesn't make sense for templates

9. Populate working primer plate info
    1. Create `process` and `primer_working_plate_creation_process` records for working plate creation

        - all steps below this can be created automatically by lab manager software in the future when new working plates are made; manual creation is necessary only for those currently in use
        - currently have `primer_set_id` on `primer_working_plate_creation_process`; could theoretically be in conflict with `primer_set_id` on `primer_set_composition` as db itself does not enforce consistency

    2. Create `plate` record for each working plate in the primer set
    3. Create `container` and `well` records for each well on each working plate

        - `container`'s `latest_upstream_process_id` will be generic `process` for `primer_working_plate_creation_process`

    4. Create `composition` and `primer_composition` records for contents of each well on each working plate

        - `composition`'s `upstream_process_id` will be generic `process` for `primer_working_plate_creation_process`

16S Run

1. Preparation

    1. Create `reagent_composition` record for each reagent to be used in process

        - Could also be done just before use, if necessary/preferred

2. Sample plate creation
    1. Create `process` record for sample_plating

        - no need for a specific process subtype here unless there are details you want to capture

    2. Create `plate` record for each new sample plate
    3. Create `container` and `well` records for each well on new sample plate

        - `latest_upstream_process_id` will be `process` id for sample_plating

    4. Create `composition` and `sample_composition` records for contents of each well of sample plate

        - `upstream_process_id` will be `process` id for sample_plating
        - `content_type_id` will be sample, blank, etc, and `content_id` will be, for example, a given sample's sample_id in Qiita

3. gDNA extraction
    1. Create `process` and `gdna_extraction_process` records for gdna extraction process
    2. Create `plate` record for each new gdna plate
    3. Create `container` and `well` records for each well on new gdna plate

        - `latest_upstream_process_id` will be generic for `gdna_extraction_process`

    4. Create `composition` and `gdna_composition` records for contents of each well of gdna plate

        - `upstream_process_id` will be generic for `gdna_extraction_process`

4. 16S library preparation
    1. Create `process` and `16s_library_prep_process` records for library prep
    2. Create `plate` record for each new library prep plate

        - Note: if I misunderstand, and prep is done in the plate produced by the gdna extraction, then new plates would NOT be created--this step would be skipped

    3. Create `container` and `well` records for each well on new library prep plate\

        - Note: if I misunderstand, and prep is done in the plate produced by the gdna extraction, then new plates would NOT be created--this step would be replaced by simply updating the `latest_upstream_process_id` to generic for `16s_library_prep_process`

4. Create `composition` and `16s_library_prep_composition` records for contents of each well of library prep plate

    - this happens even if same plate as gdna extraction is used--the CONTENTS of the wells would now different, so new compositions would be necessary
    - `upstream_process_id` will be generic for `16s_library_prep_process`

5. Quantification
    1. Create `process` and `quantification_process` records quantification process

        - could get away without `quantification_process` table since it holds no special info; I only have it in there to enforce that `concentration_calculation` and `normalization_process` must be linked to *quantification* process and not some other kind. Could maybe enforce this with a trigger instead

    2. Create `concentration_calculation` record for contents of each well that is quantified

        - `upstream_process_id` is NOT generic; is directly `quantification_process` id
        - `quantitated_composition_id` is generic for relevant `16s_library_prep_composition`

            - must be generic because same `concentration_calculation` table holds other measurements, e.g. of `shotgun_library_prep_composition`s for shotgun process

        - Note that the `upstream_process_id` on each `composition` being quantified, and the `latest_upstream_process_id` on each `container` containing the stuff being quantified is NOT updated, because **the act of quantifying the contents does not change them**

6. Plate pool creation
    1. Create `process` and `pooling_process` records for pooling process
    2. Create `container` and `tube` records for each new plate pool

        - `latest_upstream_process_id` will be generic for `pooling_process`

    3. Create `composition` and `pool_composition` records for each library prep plate being pooled

        - `upstream_process_id` will be generic for `pooling_process`
        - `container_id` will be container_id of the new tube

    4. Create `pool_composition_component` record for contents of each well in each library prep plate being pooled

        - `output_pool_composition_id` NOT generic; is directly `pool_composition` id
        - `input_composition_id`s are generic of each `16s_library_prep_composition`

7. Sequencing pool creation
    1. Create `process` and `pooling_process` records for pooling process
    2. Create `container` and `tube` records for new sequencing pool

        - `latest_upstream_process_id` will be generic for `pooling_process`

    3. Create `composition` and `pool_composition` records for new sequencing pool

        - `upstream_process_id` will be generic for `pooling_process`
        - container_id will be container_id of the new tube

    4. Create `pool_composition_component` record for contents of each well in each library prep plate being pooled

        - `output_pool_composition_id` NOT generic; is directly `pool_composition` id
        - `input_composition_id`s are generic of `pool_composition` for each plate pool

8. TODO: create run info record

Shotgun Run

Same as 16S run up through step 3 ("gDNA extraction"), then followed by 16S run step 5 ("Quantification"), except read `gdna_composition` for `16s_library_prep_composition`.

1. Normalization
    1. Create `process` and `normalization_process` records for each gdna plate being normalized

        - `quantitation_process_id` is NOT generic; is directly `quantification_process` id

    2. Create `plate` record for each new normalized gdna plate
    3. Create `container` and `well` records for each well on new normalized gdna plate

        - `latest_upstream_process_id` will be generic for `normalization_process`

    4. Create `composition` and `normalized_gdna_composition` records for contents of each well of new normalized gdna plate

        - `upstream_process_id` will be generic for `normalization_process`

2. Shotgun library preparation
    1. Create `process` and `shotgun_library_prep_process` records for each normalized gdna plate being prepped
        - `normalization_process_id` is NOT generic; is directly the `normalization_process` id
    2. **Update** `latest_upstream_process_id` of each `container` for each well of normalized gdna plate being prepped to generic of `shotgun_library_prep_process`
        - Note that no new `plate`, `well`, or `container` records are created!
    3. Create `composition` and `shotgun_library_prep_composition` records for contents of each well of normalized gdna plate being prepped
        - `upstream_process_id` will be generic for `shotgun_library_prep_process`

Now perform 16S run step 5 ("Quantification"), except read `shotgun_library_prep_composition` for `16s_library_prep_composition`. Then return to 16S run at step 6 ("Plate pool creation") and follow remainder of 16S steps, but read `shotgun_library_prep_composition` for `16s_library_prep_composition` throughout.

Query Use-Cases

1. What is in this tube I'm holding?
    1. Look up the tube's `external_identifier` in `tube`
    2. Look up `tube`'s `container_id` in `container`
    3. Look up the `container`'s `container_id` and `latest_upstream_process_id` (as `upstream_process_id`) in `composition`
    4. If you want to know more about the details of the composition, check its `composition_type_id` and look up the `composition_id` in the subtype composition table for that composition type
        - if you have a tube, at the moment this will *de facto* be a `pool_composition`, so look up `composition_id` in `pool_composition`, then look up `pool_composition_id` (as `output_composition_id`) in `pool_composition_components` to find `input_composition_id`s for everything that went into it
        - If you then follow *those* `composition_id`s back into the `composition` table and then to their relevant subtype composition tables, you should be able to follow the make-up of those contents all the way back to the sample plate
        - Example: `input_composition_id` holds `composition_id` that leads to `shotgun_library_prep_composition_id` that links to

`normalized_gdna_composition_id` that links to `gdna_composition_id` that links to `sample_composition_id`

2. What is in this plate I'm holding?
    1. Hah, this is a trick question! Plates are just aggregations of wells, and *every well could be holding something from a completely different source!*
    2. So ... ask about a particular well instead

3. Ok, what is in well B6 of this plate I'm holding, smarty-pants?
    1. Look up the plate's `external_identifier` in `plate`
    2. Look up `plate_id`, "B", and "6" in `well`
    3. Look up `well`'s `container_id` in `container`
    4. Then start at substep 3 of use-case 1 ("What's in this tube I'm holding?") and proceed as described there

4. What primer set was used for this pool?

    1. Again, this is sort of a trick question, because there's no *a priori* reason why all the primers used have to come from the same set, but hey, let's roll with it

        - **CAVEAT:** the only way to be sure all primers used for every input come from the same set would be to look them all up

    2. Assuming the pool is in a tube, complete the steps for use-case 1 ("What's in this tube I'm holding?")
    3. Follow one of the `input_composition_id`s back to `composition` and on back until you reach its underlying entry in the `16S_library_prep_composition` or `shotgun_library_prep_composition` table
    4. Look up the relevant primer composition id (from `primer_composition_id`, `i5_primer_composition_id`, or `i7_primer_composition_id`) in `primer_composition`
    5. Look up the `primer_set_id` for that `primer_composition` in `primer_set` and read the human-readable `external_identifier`

5. What primer working plate was used for this pool?

    1. Complete the steps 2-4 for use-case 4, keeping in mind the caveat
    2. Look up the `primer_composition`'s `composition_id` in `composition`
    3. Look up the `composition`'s `container_id` in `container`
    4. Check the `container`'s `container_type_id` and look up the `container_id` in the subtype container table for that container type

        - Since as far as I know primers only hang around in plates, the relevant container is *de facto* going to be a well. Look up the `plate_id` for that `well` in `plate` and read the human-readable `external_identifier`

6. Ok, but what masters was this working plate made from?

    1. Complete steps 1 and 2 of use-case 5 ("What primer working plate was used for this pool?")
    2. Find the `upstream_process_id`

        - You could theoretically look this id up in the `process` table, find the relevant `process_type`, and look up the `process_id` in the subtype process table for that process type
        - However, for a primer working plate, the `process_type` for this *better* be `primer_working_plate_creation`, so look ...

    3. Look up the `upstream_process_id` (as `process_id`) in `primer_working_plate_creation_process` and find the `master_set_order_number`

7. This primer working plate is all screwed up; who do I blame?

    1. Complete steps 1-2 of use-case 6 ("Ok, but what masters was this working plate made from?")
    2. Look up the `upstream_process_id` (as `process_id`) in `process`
    3. Find the `run_personnel_id` for the no-good who made this working plate

8. I quantified this 16s library prep plate twice; which set of measurements did I actually use in the plate pooling step?

    1. Again, this is sort of a trick question, because there's no *a priori* reason why all the pooling components have to be based on measurements from a single quantification run, but hey, let's roll with it

        - **CAVEAT:** the only way to be sure all measurements used come from the same quantification run would be to look them all up

    2. Assuming you start with a tube containing the pool for a single plate, follow steps 1-3 of use-case 1 ("What is in this tube I'm holding?")
    3. Look up the `upstream_process_id` of the plate pool composition in the `pooling_process` table

        - Note that this is like step 2 of use-case 6 ("Ok, but what masters was this working plate made from?"): you *could* do the extra step of looking the `upstream_process_id` up in `process` and finding out what `process_type_id` it has, but if you're looking at a pool, well, it better be a `pooling_process`

4. Look up the `quantification_process_id` in the `quantification_process` table
   - if you want to know what the concentration values it produced were, get all records from the `concentration_calculation` table that have that `quantification_process_id`
5. Look up the `process_id` from the `quantification_process` table in the `process` table to find out when the particular quantification run was performed and by whom

9. I got interrupted while prepping this shotgun plate, and I can't remember where I was in the process. Did I already library prep it, or just normalize it?
   1. See use-case 2 ("What is in this plate I'm holding?")
   2. But carrying on anyway, follow steps 1-3 of use-case 3 ("Ok, what is in well B6 of this plate I'm holding, smarty-pants?")
   3. Look up `container` 's `latest_upstream_process_id` in `process`
   4. Look up `process` 's `process_type_id` in `process_type` to find out if it was `gdna_normalization` or `shotgun_library_prep`