

Инвентаризация, dev

Инвентаризация: DEV: Расписания

Для себя схема работы класса расписаний, чтобы была дока и не запутаться

Schedules

Класс расписания

- **id** - ключ
- **parent_id** - родительское расписание
- **override_id** - перекрываемое расписание (это расписание является периодом-исключением в другом расписании)
- **name** - ну как-то без имени не получается
- **start_date** - дата начала действия расписания
- **end_date** - дата окончания действия расписания
- **description** - короткое пояснение
- **history** - неограниченные по размеру заметки

Ну а где собственно время то?. Ниже

Schedules Entries

Запись в расписание. Может представлять из себя

- расписание на каждый день
- расписание на день недели
- расписание на дату
- период рабочего/нерабочего времени с даты-времени1 до даты-времени2

для этого у нас есть такие поля

- **id** - ключ
- **schedule_id** - к какому расписанию запись
- **date** - дата на которую делается запись времени работы. может иметь значение
 - def - расписание на каждый день
 - 1-7 - день недели начиная с понедельника
 - YYYY-MM-DD - расписание на конкретную дату
 - YYYY-MM-DD HH:MM:SS - начало периода неработоспособности (или наоборот включения работы), если эта запись не расписание на день, а запись о «периоде» непрерывной работы/отключения (`is_period`)
- **date_end** - окончание периода работы/неработоспособности (если `is_period`)
- **schedule** - текстовое расписание в формате HH:MM-HH:MM,HH:MM-HH:MM.
Также к каждому периоду работы можно добавлять «метаданные» в JSON формате, например: 08:00-12:30{«user»:«musaev.al»},12:30-17:00{«user»:«bardina.m»}.
потом можно будет через API запрашивать текущие (из активного в настоящее время

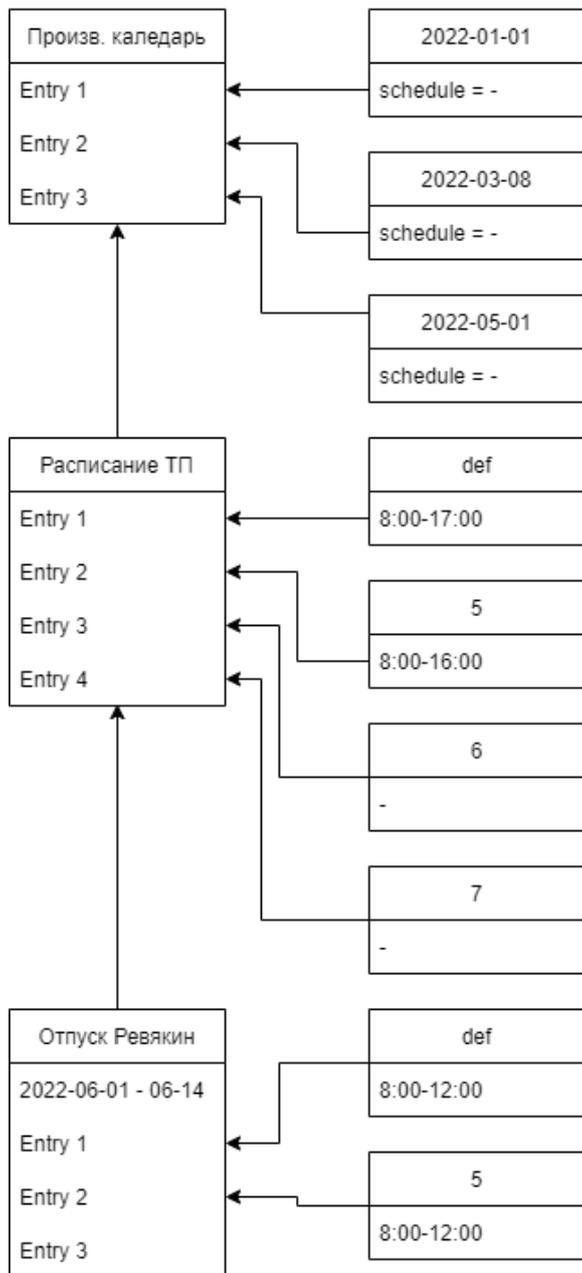
интервала) метаданные из расписания X или метаданные за период который будет активен следующим (если сейчас нерабочее время)

Отдельным валидным значением является прочерк/минус «-», означающий что в этот день рабочего времени нет.

- **is_period** - флаг того что эта запись не расписание на день, а период включения/выключения
- **is_work** - переключатель типа периода (рабочий/нерабочий), если выставлен is_period
- **comment** - короткий комментарий к записи
- **history** - неограниченные по размеру заметки

И как это все связано

попробуем зарисовать



Родительское расписание

К примеру производственный каледарь. В нем нет расписания как такового, но в нем есть **исключения**, которые будут наследоваться во все дочерние. Все унаследованные от него расписания будут иметь нерабочие дни на НГ, 8 марта и 1 мая

Дочернее расписание

К примеру график работы техподдержки

В нем определяем расписание на каждый день+
пт - сокращенный день
сб,вс - выходной

В результате имеем график работы + праздничные дни транслируемые на все расписания, которые опираются на произв. каледарь

Естественно тут тоже можно делать свои исключения, которые будут только для этого расписания и дочерних

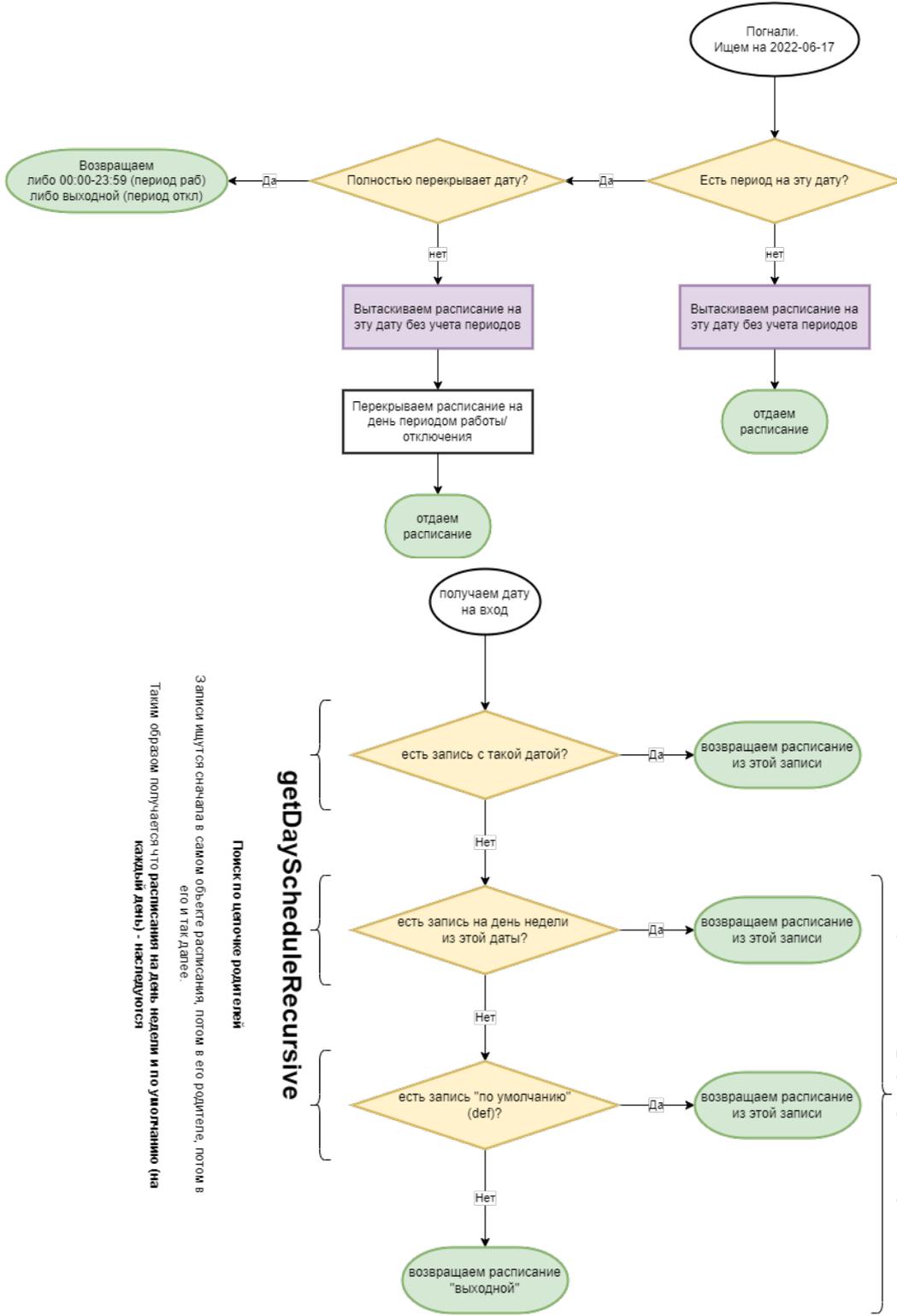
Расписание - перекрытие

Набор базовых записей расписания (ежедн + дни недели) перекрываются на какой-то период дат.

Исключения сюда не пишутся. Пишутся в исходное расписание

Периоды также сюда не пишутся, а пишутся в исходное

Определение расписания на дату



Поиск с учетом периодов

Периоды смотрим только в этом расписании, в родительские не заглядываем т.к. периоды включения-выключения не наследуются

Записи ищутся сначала в самом объекте расписания, потом в его родителе, потом в его и так далее.
 Таким образом получается, что расписание на день, неделю и по умолчанию (на каждый день) - наследуются

Поиск по цепочке родителей

getDaysScheduleRecursive

getWeekDaysScheduleRecursive

Ищет рекурсивно расписание на день недели, если не находит то исп. расписание по умолчанию

getDateScheduleRecursive

Ищет расписание на дату, сначала как явное исключение потом из расписания на неделю

