Создатель Грищенкова Дарья Павловна, отредактировано дек 07, 2023 · Add Approval · 3,135 views by 209 users since 28 Apr 2023

Компетенция	Краткое описание	Как проявляется? (конкретные действия, навыки, умения)				
Способность использовать знания, умения и навыки в конкретной деятельности (проявляется на уровне поведения: делает или не делает)	Про что?	1	2	3	4	5
Общие навыки	Навыки, общая компьютерная грамотность	Знать стандарты чаптера и прикладывать максимальные усилия для их соблюдения на проекте	Уметь объяснять различия семантик доставки данных "at-least-once", "at-most-once", "exactly-once"	Уметь реализовать доставки кроме exactly once		Уметь реализовать доставку данных по семантике exactly-once или объяснить почему этого сделать нельзя для конкретных технологий
Spark						
Знание API	Навыки работы с основным рабочем инструмент - фреймворк Apache Spark	Уметь написать код, который позволяет - посмотреть состав таблиц в базе Hive - обращаться к данным через Spark SQL-запрос - использовать global temporary view - создать SparkSession Уметь объяснить теоретические концепции Spark: - driver - executor - transformation - action	Уметь применять Spark API (Dataframes) - прочитать данные из таблицы в Hive - записать данные в таблицу в Hive - сделать промежуточные преобразования в Dataframe/Dataset API (select, map, flatMap, agg, ...) - применить библиотеку typesafe для чтения конфигурации - Уметь работать со схемой данных, проводить конвертацию типов Уметь объяснить теоретические концепции Spark: - shuffling - data locality	Уметь работать с форматами данных: avro, parquet, json, csv, orc - читать таблицы - читать файлы - писать таблицы - писать файлы Уметь работать с источниками данных, отличными от Hive: - jdbc - s3 Использовать - broadcast - accumulator - cache, persist - coalesce, repartition	Уметь запускать обработку данных в многопоточном режиме Уметь читать и записывать данные из/в Kafka Уметь делать stateful и stateless трансформации	Уметь - применять оконные аггрегации - использовать чекпойнты - использовать водяные знаки Уметь обеспечивать отказоустойчивость и заданную семантику доставки данных в потоковом приложении
Умение работать с локальными и продуктивно эксплуатируемыми приложениями	Навыки работы с основным рабочем инструмент - фреймворк Apache Spark	Запуск spark-shell, умение выполнять базовые команды Работа со Spark-проектом в IDE - выгрузить из Git и собрать готовый проект в IDE - собирать fat-jar через sbt-плагин assembly - запустить Spark-задачу локально в Intellij IDEA	Уметь запускать задачу на кластере через spark-submit, уметь объяснять отличия режимов запуска Создать и собрать проект в командной строке sbt или из IDE Уметь смотреть логи приложения через SparkUI и Yarn	Настраивать память и ядра драйвера и екзекьюторов: - heap - offheap - распределение по областям - количество и аллокацию контейнеров	Уметь настраивать произвольные параметры спарк-сессии, знать основные полезные параметры Уметь конфигурировать уровень параллелизма, объяснять как он работает Уметь читать explain и dag, находить и решать проблемы производительности задачи Уметь - запускать простые задачи на Spark Structured streaming	- уметь работать с различными OutputMode - уметь читать и объяснять Streaming Query UI
Scala						
Конструкции языка	Основной язык программирования	- основные структуры данных языка (коллекции), их виды, их различия - базовые операции над коллекциями (функции map, filter ……), уметь объяснить результаты работы данных функций на разных коллекциях - уметь работать с Tuple - уметь написать код с простейшими операциями	- уметь работать с кейс-классами, уметь объяснить преимущества перед кортежами - уметь использовать операцию foldLeft - уметь использовать for comprehension - уметь использовать “ленивые” вычисления	- уметь объяснить и описать варианты использования pattern matching, - уметь объяснить понятие implicits - уметь рассказать о монадах Try (Succes/Failure) или Option/Either, как с их помощью реализуется обработка	- уметь объяснить ко/контр/инвариантность по типу - уметь продемонстрировать в коде следствия использования чего-нибудь одного: ко/контр/инвариантности по типу - уметь объяснить карирование функций - уметь пользоваться параллельными коллекциями	- уметь пользоваться Future - разбор нетривиального случая по работе компилятора (кейс) - следствия из реализации системы типов (выведение типов) - объяснить каким образом были присвоены те или иные типы объектам,
Сборка и библиотеки	Основной язык программирования	- собирать готовый проект на sbt или maven	- создать проект из шаблона на sbt или maven	- поддерживать сборку проекта в актуальном состоянии, следить за версиями компонентов, своевременно исключать неиспользуемые зависимости из скоупа проекта - перечислить какими пользовался библиотеками - использовать инструменты отображиения дерева зависимостей	- перечислить для каких целей могут быть использованы плагины, назвать какие использовал в работе - уметь объяснять стратегии разрешения зависимостей - уметь исключать дублирующиеся библиотеки при помощи стратегий разрешения зависимостей, объяснить, что такое Jar Hell и спобобы борьбы с ним	- уметь собирать проект любым сборщиком - sbt, maven, gradle - уметь использовать плагин Shadow Jar
Паттерны	Основной язык программирования	уметь объяснять принципы ООП: абстракция, инкапсуляция, наследование, полиморфизм, уметь применять в коде принципы ООП уметь планировать необходимые объекты и их сигнатуры под задачу уметь использовать traits	- перечислить какие известны категории паттернов (ООП или ФП), назвать конкретный паттерн одной из категорий, объяснить его суть - рассказать о функциях высших порядков, продемонстрировать реализацию любой функции высшего порядка, объяснить возможную пользу	- рассказать какие алгоритмы могут быть реализованы более просто или более эффективно на scala чем в декларативном стиле - объяснить сферу разумного применения implicits, продемонстрировать в коде работу неявных преобразований, если использует в работе - аргументировать необходимость использования данной возможности	- архитектура приложения в функциональном стиле - объяснить какие отличия от проектирования решений в декларативном стиле, какие возможности дает и ограничения накладывает	- знать паттерны функционального программирования - работать с актуальными фреймворками/либами (akka/play/zio/cats etc.)
Python	Навыки анализа и обработки информации с помощью python	- уверенное знание синтаксиса Python - использовать PEP8/Zen of Python - знает основные типы данных, циклы, условия, операторы, работа с файлами - знать изменяемые и неизменяемые типы данных - понимать особенности и ограничения базовых структур данных (set, dict, list, tuple и др.), применяет их по назначению - импорт библиотек	- уметь использовать virtualenv для изоляции зависимостей - знать о comprehensions, преимущества их использования - знать что такое упаковка и распаковка, *args, **kwargs - знать что такое декораторы, декораторы с параметрами, знание functools.wraps - знать что такое итераторы и генераторы, какая между ними разница - ссылки и копии - знать что такое контекстный менеджер, уметь реализовать - PYTHONPATH - документация кода, аннотация типов	- концепции ООП (уметь создать класс, уметь создать методы и экземпляры класса, наследование классов, вызов методов суперкласса) - уметь покрывать код тестами, использовать фреймворки для тестирования unittest, pytest, fixtures, parametrize - уметь обрабатывать исключения, вызывать собственные исключения - знать функции высших порядков: map, filter, reduce	- уметь определять bottle necks в приложении, профилировать, оптимизировать приложения в плане производительности - GIL, multithreading, multiprocessing - магические методы	- разбираться в устройстве виртуальной машины, GC - уметь работать с фреймворками: FastApi, Flask, Django, Flask или другими - умеете работать с синхронными и асинхронными фреймворками, AsyncIO - умеете работать с реляционными СУБД - знать принципы RESTful API
IDE (Intellij Idea, если используется другая на проекте - вопросы будут про неё)	Умение использовать инструменты разработки	Уметь открыть, собрать проект, скачать изменения из Git, запушить свои изменения в feature ветку Уметь перемещаться по коду с использованием мыши Уметь запустить код локально	Уметь устанавливать и использовать плагины для разработки Уметь менять настройки конфигураций запуска Уметь отлаживать код через IDE Уметь создать проект из готового шаблона Знать комбинации клавиш для автоматического форматирования кода и оптимизации импортов	Знать основные горячие клавиши для навигации по IDE, для сборки проекта и для работы с Git Выполнять статический анализ кода и замер покрытия тестами из IDE Уметь использовать генераторы кода, встроенные в IDE Уметь настраивать профиль для автоматического форматирования кода и импортов	Уметь создавать шаблоны проектов Уметь настраивать параметры самой IDE для эффективной работы	Следит для новыми трендами в области разработки IDE, предлагает/пишет новые плагины для ускорения работы
Airflow	Умение использовать инструменты запуска задач по расписанию	Создавать DAG и документировать их Настраивать периодичность запуска DAG Использовать SparkSubmitOperator, BashOperator, PythonOperator Пользоваться UI Airflow Перезапускать DAG вручную	Пользоваться другими операторами Airflow Получать логи Airflow (OpenSearch) Знать основные атрибуты у объекта DAG (catchup, depends_on_past e.t.c) Уметь использовать Variables Уметь использовать XCOM Уметь объяснить что такое Trigger Rules	Применять сенсоры и макросы Смотреть и анализировать статистику запуска DAGов Уметь подключать новые библиотеки операторов Уметь пользоваться командной строкой при работе с Airflow	Создавать тесты для DAG-ов Знать архитектуру Airflow	Создавать плагины для Airflow Создавать свои custom операторы
Oozie	Умение использовать инструменты запуска задач по расписанию	Зачем нужен Oozie Создавать, запускать, останавливать workflow, coordinator из шаблонов в командной строке Смотреть логи задач Писать линейные воркфлоу со стандартными действиями	Писать воркфлоу с ветвлениями и циклами Использовать EL операции	Обмениваться данными между действиями в workflow, обеспечивать корректное завершение работы workflow и логирование ошибок в случае отказа	Создавать тесты для workflow	Расширять функциональность Oozie
NI-FI (опционально)		знать • что такое NiFi, для чего используется • что такое процессор в NiFi • как соединяются процессоры • какие у них есть базовые свойства уметь • выполнять навигацию по NiFi • создавать flow • сохранять flow в NiFi Registry • открывать и редактировать flow • настраивать расписание запуска процессора и количество ресурсов	знать • как можно управлять процессорами и flow вручную • чем отличается Terminate от Stop • типы процессоров по их предназначению • базовый workflow NiFi-разработчика • что такое атрибуты • что такое custom-свойства • для чего процессоры группы Attribute Extraction • что из себя представляет язык выражений в NiFi • помнить основные методы языка выражений уметь • запускать, останавливать отдельные процессоры • запускать, останавливать весь flow целиком • использовать базовые типы процессоров: DataTransformation, Routing and Mediation, Database access, Splitting and Aggregation • создавать пайплайны из готовых процессоров в соответствии с базовым workflow разработчика - от начала до мониторинга в Графане • задавать и читать значения атрибутов и свойств в NiFi-flow • уметь использовать процессоры для Attribute Extraction • практически использовать язык выражений в NiFi-процессорах	знать • что такое мониторинг • что такое data provenance в NiFi • какие статистики и графики доступны • как вызывать внешние скрипты в NiFi • что входит в состав группы System interaction уметь • пользоваться мониторингом • пользоваться data provenance в NiFi • смотреть статистики • вызывать внешние скрипты в NiFi • пользоваться процессорами группы System Interaction	знать • как NiFi умеет взаимодействовать с Kafka • как NiFi умеет взаимодействовать с Hive, Hadoop • как NiFi умеет взаимодействовать с HTTP, файловой структурой уметь • делать flow, которые получают/отдают данные из Kafka • делать flow, которые получают/отдают данные из Hive, Hadoop • делать flow, которые получают/отдают данные из HTTP, файловой структуры	знать • порядок действий чтобы разработать, собрать и запустить свой собственный процессор • какие кастом-процессоры используются в инфраструктуре уметь • создавать nar-бандлы, тестировать их и выводить в продуктив • разбираться в коде уже существующих кастом-процессоров
SQL		уметь делать простые выборки из одной таблицы уметь фильтровать строки таблицы по условию уметь применять агрегатные функции уметь фильтровать результат выборки по результатам получаемого значения агрегатной функции уметь работать с NULL значениями	уметь объяснить результат применения разных видов join между таблицами уметь выполнять вставку данных уметь выполнять обновление данных уметь выполнять удаление данных уметь работать с вложенными запросами	уметь создавать таблицы разными способами уметь изменять таблицы (название, структура) уметь удалять таблицы (данные в таблице) уметь создавать, убирать ключи знать виды партицирования таблиц	уметь создавать представления (view) уметь создавать материализованные представления уметь пользоваться оконными функциями в запросах уметь пользоваться ROLLUP, CUBE	план запроса способы join:Nested loop , Hash join, Merge join способы оптимизации запросов хинты иерархические запросы триггеры
Bash/Linux		уметь переходить по директориям уметь просматривать содержимое директории уметь создавать и удалять директории уметь создавать и удалять файлы	уметь выдавать права на директории и файлы уметь изменять возможность исполнения файлов уметь искать файлы уметь пользоваться одним из консольных редакторов файлов - vim, nano, ...	уметь выводить содержимое файлов на экран (целиком, первые строки, последние строки) уметь пролистывать содержимое больших файлов уметь выводить на экран отфильтрованное содержимое файла по наличию ключевых слов в строках уметь писать свои bash-скрипты	Знать особенности работы Bash скриптов, Уметь менять настройки, использовать параметры Уметь использовать служебные параметры bash скрипта ($$, $?, и т.д.) Использовать cron	Знать базовые конфигурации системы, уметь их настраивать уметь использовать sed уметь использоваться awk
HDFS		Уметь просматривать список файлов на HDFS Уметь загружать файлы на HDFS Уметь скачивать файлы с HDFS	Уметь выполнять базовые команды hdfs в CLI Уметь выдавать права на файлы и директории Знать архитектуру HDFS, уметь объяснить как работают основные компоненты HDFS	Уметь выводить содержимое файлов на экран Уметь определять оптимальные размер файла на HDFS и контролировать его		
CI/CD		Знает основные команды cli Уметь запускать docker-контейнеры и делать базовые операции с ними Знает что такое Docker Registry	Умеет конфигурировать docker-compose.yml Умеет создавать новые образы контейнеров на основе других образов Умеет работать с Docker Hub	Знает основы Hashicorp Vault Умеет настраивать взаимодействие с vault Умеет создавать новые образы контейнеров без использования других образов	Умеет настраивать deploy в gitlabRegistry Умеет использовать шаблоны CI/CD для конфигурации пайплайнов в своих проектах	Умеет настраивать delpoy в dev/prod namespace k8s Умеет настраивать ingress
Логирование/ Мониторинг		Умеет пользоваться логгером в коде приложения, Умеет смотреть логи работающего приложения	Умеет логировать предупреждения, ошибки при выполнение кода. Умеет настраивать уровни логирования приложения в коде, при запуске. Умеет смотреть логи завершившего работу приложения в системе сбора логов	Умеет отправлять и получать метрики из VictoriaMetrics Умеет создавать графики в Grafana Умеет настраивать алертинг в мессенджеры и почту из Grafana	Умеет организовать сквозной мониторинг бизнес-процессов в приложении. Например, уметь отразить в мониторинге актуальность данных отчета, базирующегося на других отчетах Умеет настраивать DQ	Умеет подобрать компоненты и настроить кросс-системный мониторинг Например, если приложение копирует данные из СУБД в HDFS, уметь отразить работоспособность всех узлов, качество доставки данных (соответсвует ли заявленной семантике) и задержку в доставке данных от момента их рождения в системе-источнике
Знание предметной области	уникальные экспертные знания по конкретным проектам, бизнес системам и элементов сети	обладает минимальными знаниями проекта/продукта	обладает базовыми знаниями проекта/продукта	Обладает уверенными знаниями проекта/продукта. Делится знаниями с коллегами из других команд	Обладает высокими знаниями проекта/продукта. Проактивно предлагает новые идеи для развития продукта Делится знаниями с коллегами внутри команды и иногда во вне Проводит встречи и митапы с коллегами	Обладает экспертными знаниями и уникальными знаниями продукта. Проактивно делится знаниями, описывает процессы и помогает в развитие продуктаелится знаниями и экспертизой с коллегами внутри и во вне. Активно участвует в обсуждениях с бизнес заказчик
Коммуникационные навыки	Взаимодействие с другими участниками процесса разработки, заказчиком, внешними партнерами и продуктовыми командами	Не взаимодействует с членами других команд Редко взаимодействует с членами своей команды, пассивно ведет коммуникацию	Редко взаимодействует с коллегами из других команд, основная коммуникации идет внутри команды-общается вежливо, своевременно отвечает на сообщения	Постоянно применяет индивидуальный подход при выстраивании коммуникации, взаимодействует с разработчиками и аналитиками из других команд -общается вежливо, своевременно отвечает на сообщения	Способен найти общий язык с любым сотрудником, общается вежливо, своевременно отвечает на сообщения -активно участвует в обсуждениях вне рамок команды и департамента. Предлагает решения -выступает на митапах и внешних конференциях	Организует встречи с экспертами для обсуждения задач. Обладает навыками управления коммуникацией, Умеет заинтересовать участников встречи Помогает найти общий язык членам своей команды, может выступить посредником при налаживании коммуникации и способен урегулировать конфликт Выступает на митапах, внешних конференциях
Уровни решаемых задач		Алгоритмическая постановка задачи, способен по четким шагам внести изменения в существующую кодовую баз, проверить вручную работоспособность приложения и создать мерж реквест	Предметная постановка задачи, способен по четкому описанию результата в технических терминах, в заданном стеке создать с нуля или существенно переработать существующее приложение, вручную и автоматическими средствами проверить корректность решения, вывести его в продуктив и обеспечивать функционирование	Предметная постановка задачи, способен по четкому описанию результата в технических терминах, в заданном стеке создать с нуля или существенно переработать существующее приложение, вручную и автоматическими средствами проверить корректность решения, вывести его в продуктив и обеспечивать функционирование	Способен формализовать нечёткие ожидания заказчика вместе с ним, предложить и реализовать путь решения задачи, обеспечивающий максимальную прозрачность и скорость достижения результата	Способен проводить доказательные исследования для выбора инструментов, внедрять их в экосистему и процессы компании от начала до конца
Ответственность и принятие решений		Ответственность в рамках текущих небольших задач (проверить гипотезу, собрать фичи, выгрузить данные) Влияние в пределах краткосрочных задач внутри продукта	Ответственность в рамках длительного трэка задач Влияние в пределах длительного трэка задач внутри продукта	Ответственность в рамках нескольких трэков задач внутри продукта Влияние в рамках нескольких трэков внутри, включая внутрикомандные взаимодействия Составляет краткосрочные роадмапы, эффективно выстраивает внутрикомандные взаимодействия Сосредоточен на внедрении инновационных подходов для улучшения качества продукта	Ответственность за стратегию развития всего инженерного продукта и его эффективность Влияние в масштабах всего продукта, а также соседних зависимых, включая кросскомандные взамодействия Влияние на стратегию развития чаптера Составляет долгосрочные роадмапы	Ответственность за инженерную стратегию нескольких сложных продуктов Влияние в рамках бизнес-юнита и чаптера

Нравится · Станьте первыми кому понравится это · Нет меток

Написать комментарий...