**Задача 1**

Архитектор ПО решил проконсультироваться у вас, какой тип БД лучше выбрать для хранения определенных данных.

Он вам предоставил следующие типы сущностей, которые нужно будет хранить в БД:

* Электронные чеки в json виде
* Склады и автомобильные дороги для логистической компании
* Генеалогические деревья
* Кэш идентификаторов клиентов с ограниченным временем жизни для движка аутенфикации
* Отношения клиент-покупка для интернет-магазина

Выберите подходящие типы СУБД для каждой сущности и объясните свой выбор.

Ответ:  
1.1. Электронные чеки в json виде

Если такие данные требуется быстро выдернуть и передать или изменить на лету, то система ключ-значение.  
Либо документо-ориентированные если нужно хранить долгосрочно данные.

1.2. Склады и автомобильные дороги для логистической компании  
Иерархическая БД (склад – маршрут автомобильные дороги). Если строить на платных, как я понял из лекции это продукт MySQL Oracle.

либо сетевая БД, как я понимаю OpenSource PostgreSQL.  
  
1.3. Генеалогические деревья

Сетевая БД, предполагаю что проще это все организовать на NoSQL решениях.

1.4. Кэш идентификаторов клиентов с ограниченным временем жизни для движка аутенфикации.

Ключ-значения - применение этих баз данных это прогретые кеши (Memcached и Redis), которые хранятся в оперативной памяти и которые нужны для того чтобы хранить там не самые большие значения.

1.5. Отношения клиент-покупка для интернет-магазина

Графовые. NoSQL системы. Т.к. тут разные клиенты собирают разные товары (используются разные ID). Ставятся все эти лайки и используются различные связи между с уникальными свойствами; могут выстраиваться ребра –отношения которые имеют свои свойства.

**Задача 2**

*Вы создали распределенное высоконагруженное приложение и хотите классифицировать его согласно CAP-теореме. Какой классификации по CAP-теореме соответствует ваша система, если (каждый пункт - это отдельная реализация вашей системы и для каждого пункта надо привести классификацию):*

Данные записываются на все узлы с задержкой до часа (асинхронная запись)

При сетевых сбоях, система может разделиться на 2 раздельных кластера

Система может не прислать корректный ответ или сбросить соединение

А согласно PACELC-теореме, как бы вы классифицировали данные реализации?  
  
2.1. Данные записываются на все узлы с задержкой до часа (асинхронная запись)  
  
доступность (англ. availability) — Любой запрос к функционирующему узлу распределённой системы завершается корректным откликом, однако без гарантии, что ответ содержит наиболее актуальные данные  
**по PACELS**: PA/EL Dynamo, Cassandra

2.2.При сетевых сбоях, система может разделиться на 2 раздельных кластера

partition tolerance — устойчивость к разделению расщепление распределённой системы на несколько изолированных секций не приводит к некорректности отклика от каждой из секций.  
**по PACELS**: PA/EC MongoDB  
  
2.3 Система может не прислать корректный ответ или сбросить соединение  
Сonsistensy Каждое прочтение получает самую последнюю запись или ошибку. <https://habr.com/ru/company/gaz-is/blog/551986/>  
**по PACELS**: EC/PC Megastore, VoltDB  
  
Задача 3

Могут ли в одной системе сочетаться принципы BASE и ACID? Почему?

Не могут, принципы противопоставляются друг другу.

принцип ACID говорит о том, что везде получаем одинаковый ответ и все произойдет но не сразу.

BASE говорит о том что если сбой в некоторых узлах, то он не приводит к отказу в обслуживании всего сервиса, только незначительной части сессии. Данные хранятся time to live. Имеется противоричивость данных.  
  
  
Задача 4

Вам дали задачу написать системное решение, основой которого бы послужили:

фиксация некоторых значений с временем жизни

реакция на истечение таймаута

Вы слышали о key-value хранилище, которое имеет механизм Pub/Sub. Что это за система? Какие минусы выбора данной системы?  
  
<https://veesp.com/ru/blog/sql-or-nosql/>  
  
Хранилище вида “ключ-значение”

Хранилища “ключ-значение” представляют собой простейший вид базы данных, являясь, по сути, ассоциативным массивом - каждому значению сопоставляется свой уникальный ключ. Простота хранилищ этого типа открывает просторы невероятной масштабируемости. Не требуется никаких схем построения базы данных, нет никакой связи между значениями, по сути количество элементов ассоциативного массива ограничено лишь вычислительными мощностями. Именно потому данный вид хранилищ интересен в первую очередь компаниям, предоставляющим услуги облачного хостинга.

С другой стороны, простота хранилищ “ключ-значение” очень затрудняет или полностью отсекает большинство привычных операций со значениями хранилища - если ключами можно ещё оперировать как угодно, то попытка выполнить поиск по значениям может длиться на несколько порядков дольше, чем в реляционной базе данных. А вместе с ограниченным набором манипуляций над значениями ячеек хранилища идёт и фактическая невозможность быстро анализировать имеющуюся в базе данных информацию и собирать статистику.