Netflix has grown significantly over the past few years, from offering online movies to a few thousand clients to becoming a multinational firm whose streaming services account for more than 30% of the peak downstream traffic in the US. Currently, Netflix handles billions of instances of viewing data each day. However, the firm believes that it has to re- architect its systems to keep pace with the ever increasing demand. As a consequence, Netflix plans to use different databases, or storage technologies, to maintain its data. For instance, Netflix is considering the use of Cassandra—an open source, NoSQL distributed database—for promptly writing high volumes of data into storage, and Redis—another open source, NoSQL database—for rapidly reading high volumes of data. Let's assume that you are working for a software consultancy called Plymouth IT Consultants, which has just been hired by Netflix to re-architect its database systems. As Director of Engineering at Plymouth IT Consultants, you have been asked to propose a new database architecture based, entirely, on NoSQL products. Using the material introduced in the business intelligence section of the ISAD353 module as a starting point, you are to write a 3,000-word report discussing the particular NoSQL database product that you have chosen for handling Netflix's recommendation system. Essentially, you have to choose a particular type and example of NoSQL database, and justify why this choice is better than others. Note that your report is NOT required to deal with customer memberships and subscriptions, video on demand, DVD delivery by mail, or any other aspect of the Netflix business. Your report should focus, EXCLUSIVELY, on Netflix's recommendation system.

Question:

'Essentially you have to choose a particular type and example of NoSQL database and justify why this choice is better than others' - Picked graph databases, specifically Neo4j - Throughout essay, when I make a point for Neo4j, link to other NoSQL options and comapre and contrast against them - Critically analysis the pros and cons of all - Explain and give evidence for HOW it will work - weigh up - Which comes out on top - Why? - Justify answer with points above - Making sure that it is given in literature

DONT:

- reuse points of the powerpoint slides
- keep redefinitions to a minimal
- describe and explain relationship between citation and your own report
- Cite by using numbers ... [3] and then number reference list
- Analysis the reference and include by re-wording, don't use direct quotes
- Don't speculate, only present facts, explain and analysis

- ONLY RECOMMENDATION SYSTEM
- MINIMAL/LITTLE TO NONE SPECULATION
  - ALWAYS FACTS FROM LITERATURE
- NOT JUST PROS AND CONS
  - BUT HOW AND WHY

Writing prompts:

- Get every feature of graph databases
  - Link it to the question
  - compare it other NoSQL options

Think big - Multinational - How does these databases cope with huge reads and writes? - Do these databases cope as they scale

- THINK SCALABLE!
  - Know that amazon use it
  - Try to find examples with them

Have the reading and writing seperated - Over head of maintaining two databases Have the reading and writing together - No as good at performance

Need reading/writing speeds!

think first try define what types of NoSql will be useful then decide which within those categories are appropriate - Don't just make it pros and cons - How would it work? - How would you read and write data

how can data be adequately exploited within the NoSQL database to match a client's preference and ratings, viewing histor or friends' recommendations how can it find the data and use it so it can make recommendations - like, "your friend also watched this movie" Are there going to be any issues arising with the database choosen when handling data In a graph database what will the nodes represent and the relationships be? - Use examples and evidence from literature and give details of specific examples

SMALL paragraph on the benefits of NoSQL database in this area over relational

try finding multiple examples for each available option - youtube videos could be good

- This is an example for using NoSql
- *Don't assume you need two different types of databases*
- Can compare my choices against those choices
- Don't have to do Read/Write
- **Main part is recommendation systems**
- Do the rational behind the choices

Starting points:

RESEARCH Use Graph databases - look into other approaches briefly - To find out why they're not suitable - Research cassandra - Research Redis - Look at other multinational companies using graph databases like **amazon** - Neo4j - **Clustering** - Look at the use of graph databases in recommendation systems - Content based systems (item-to-item) done - collaborative filtering (user-to-user) done

GRAPH DATABASE RESEACH

Graph databases

Basic definition - Based on graph theory - says to me that things are linked and correlated - Which is linked to recommendation systems - Because if something is to be recommened it has to be linked to user and their prefrences - Have directional significance - Important is that it is easy to navigate - Very good for recommendation systems as there will be a lot of amalgamated data about a user that needs to be traversed in real time to get to the conclusion that is what is to be recommended - As it traverses it can collect information as it goes - Can it dynamically change is SQL statement to deal with new data? - If has a friend who liked this then also does he like this? - If their programming in java or scala then it will be 'easier' to integrate - Else could cause some overheads - It is the connectedness that gives it the advantage - working with data that is correlated - When data is connected it may give insight into data that was otherwise unknown - Provides personalisation that is otherwise lost - Can further use this personalisation to improve the system - Scalable and flexible - **FIND OUT HOW/WHY?** - Importand in an ever changing/growing environment

What would the relationships and nodes represent?

noun(node/entity) -> verb(relationship) -> noun(node/entity) subject -> verb -> object Joe -> owns -> toyota - this is graph theory

- Relationships:
  - Also watched
  - Same genre
  - Same rating
  - Is friends of
- Nodes:
  - People
  - Friends
  - Movies
- Further advance:

    - Using meta data
    - It would determine how long you were on a film for before switching off
    - What time of day you watch what
        * From this it could infere:
            · what you like and don't like if u turned it off early
            · What type of programs you watch at what time

Uses of Graph databases in real world:

- Walmart: http://info.neo4j.com/rs/neotechnology/images/neo4j-casestudy-walmart.pdf
- Can't get much bigger than this
    - although it is only the Brazilian section of the company it still a part of the biggest company in the world
    - Wanted it for recommendations
- Facebook: https://neo4j.com/blog/why-the-most-important-part-of-facebook-graph-search-is-graph/
    - 1000 times faster than relational database when using connected data
    - https://www.manning.com/books/neo4j-in-action

Use of graph databases with recommendation platform

https%3A%2F%2Fdiuf.unifr.ch%2Fmain%2Fis%2Fsites%2Fdiuf.unifr.ch.main.is%2Ffiles%2Fdocuments%2Fstude
projects%2FGroup_3_Cung_Jedidi.pdf **Quotes:** - "A graph data model, although being fairly new has many solutions to improve business applications that have a huge amount of data with a high degree of correlation."

Content-based system (item-to-item)

- AKA cognitive filtering
- "A model whereby the recommendations are made on the basis of serveral properties of the product"
- "Compares the similarities of the complementarity of the elements"
    - properties of the previous watches
- "The user isn't asked to put in attributes"
- recommends items based on a comparison between the content of the items and a user profile
- The content of each item is represented as a set of descriptors or terms
    - Words that discribe the document

- Problem occours when assigning these discritpors
    - Can the be done automatically or manually?
    - Costs
- there has to be a method to extract these terms
- They have to be represented such that both user profile and the items can be compared in a meaningful way
- A learning algorithm has to chosen that is able to learn the user profile based on seen items and can make recommendations based on this user profile
- **Disadvantages:**
- Scalability
- When user's actions on the system become larger the quality of the recommendation drops
    - Just too much data which is too broard to make specific recommendations to give that personilised feel *own words*
- "With only an"is similar to" relationship between products, the recommendation can be rather poor"
- "It would be more interesting to know how a product is similar to another one and the level of the similarity"
- The user's preferences also change over time and are rarely static
    - The system has to adapt
    - Either it can be:
    - **Exploitation** - The system chooses documents similar to those for which the user has already expressed a preference
    - **Exploration** - The system chooses documents where the user does not provide evidence to predict the user's reaction

Own judgements: - Easily used within netflix because it is all about what the user watches - Each of these watches will have a lot of data attached to it - Genre, similar films, similar length, actors, directors - These can work up until a point, scalbility could be the issue - The more movies that get added the less specific the recommendations become - *DO NOTICE THAT IT DOESN'T GROW SO MUCH AS DATA PUT IN SUBSTITUES FOR DATA TAKEN OUT* - User's choices and behaviours are notourisly hard to pin - Not often do you find someone who will only watch a certain type of movie, becomes very general and therefore the recommendations become more broad

Collaborative filtering (user-to-user)

- AKA social filtering
- "The system builds profiles of users based on their behaviours and compares the different user profiles to provide a recommendation"
- "The system will find correlation between users' purchases and recommend a product to a customer based on the items bought by similar users"
- "A common practice to improve this technique is to ask the user to rate the products, so that it becomes much easier to find common pattern among users"
- "By this way, the model is transformed into a sgementation model. The customer's preferences are analyzed and the recommendations are made based on the prefrences of other users of the same segment."
- "So, the users collaborate to build their 'global' profile and profit from others' ratings"
- Implicit ratings based on user's behaviour also help to give ratings (see metadata)
- Uses recommendations of other people
- Based on the idea that people who agreed in their evaluation of certain items in the past are likely to agree again in the future
- You ask for similar recommendations of off friends that have similar taste
- This information is used in the decision

Neighbourhood-based technique

- A number of users is selected based on their similarity to the active user
- A predicition for the active user is made by calculating a weighted average of the ratings of selected users
- To decide if one person would like to watch a movie
  - their ratings are compared to the ratings of ther others
  - If I liked this you will probabily like it too
- How to choose a neighbour
  - Cannot predict thousands people at runtime
  - **correlation-thresholding** only selects those neighbours who's correlation is greater than a given threshold
  - **best-n-neighbour** selects the best n neighbours with the highest correlation

Disadvantages: - Defining segments of users could lead to poor quality recommendations, since all the users of a same segment are considered to have exactly the same preferences - Doing some tweaking on the segments, like using granularity, would become expensive to just determine those segments - Collaborative filtering sustems requires time to build up profiles since it is based on past behaviours - the user slowly builds a profile which is analysed - New user preferences are difficult to determin and the strategy would be rather asking to provides inputs by himself - Ther participation of the user is crucial to determine the accuracy of the recommendations - *Sparsity problem* - To few ratings when the items become cery large, reducing the number of items users have rated to a tine percentage - in such situation it is likely that two people have few rated items in common making the correlation coefficient less reliable - *TO SOLVE:* - Implicit ratings - from behaviour picked up by meta data - dimensionality reduction - reducing the deminsionality of the information space

Item-toitem approach

- Use the neighbourhood approach but instead of correlation between user's and their preferences it is between products
- The ratings between products are used

Own judgements: - Implicit ratings based on user's behaviour also help to give ratings (see metadata) - This could be where the meta data can come in - Still properties but not properties of the user's action towards these elements - This could build up richer relationships of between the user and the product to get further insights and inturn, get better recommendations - If the user profiles take a while to build up, would this be necessary? - Is there a way in which it can use already known data, behaviours and segments to implement this seemlessly and still be able to provide solid recommendations? - What happens with new users, how can they be placed in a segment - More data attributes like location, age, sex would be used but the user has to provide these - *Don't get too hung up on the nitty details of each recommendation system*

Neo4j

- Implemented Java and Scala
- Can interact with database from Java program
  - If their product is already in Java then this is an easy integration
- Has its own language of CQL
  - Cypher query language
  - When writing it is about pattern matching
- (more training to use)
- Clustering
- Supports ACID

What information is represented

- highly interconnected data
  - Movies and all the things that come along with it, actors, directors, genres
- data that needs to be traversed efficiently

    - To provide real time recommendation

- Does well at searching patterns of interations through and between data
- Data that is increasing and the connections between data that are increasing
- To understand the need to use a graph database for a recommendation system you need to understand the significance of how connected the data is.
    - This is the graph database's one true strenght
    - Working effeiciently with connected data
        * HOW?
        * WHY?

OTHER DATABASES

Cassandra - promptly writing high volumes - Think

Redis - Reading

Writing structure

- Breif of SQL vs NoSQL in context of recommendation systems
- Introduction to graph databases and Neo4j
- Features of Neo4j
- How it works with data
    - What the data is
    - How it is connected
    - How it finds patterns with it
    - How effiecent it is at doing so (figuresss)
- Clustering
- different types of recommendation
    - content based
    - Collaborative
- what the data would be