**https://marcel.ai/classes/course/course:2545716#  
https://marcelclasses.udemy.com/course/ai-agents-for-business-automate-work-without-coding/learn/lecture/52394301#overview**

Vector DB- A tool that works with unstructured data in the form of vectors  
Compute engines for vector data  
Vectors are long serial numbers  
Generated by deep neural networks  
  
**Embedding Models**

Neural networks trained on some types of data.  
Match the data type and model type.  
Can only quantitatively compare embeddings of the same length.  
  
**Where does LLMS come in?**

LLMS are stochastic models; they predict the next token  
Trained on publicly available data  
RAG uses the LLM to make sense of your question and send that to vector database for context to answer your question.  
  
**Chunking:**Process of breaking down big blocks of text  
**Embeddings:**Vectors generated by Embedding Models, numerical representation of unstructured data, sparse and dense  
**Dense – few zeros**

**MetaData:**Data Stored along with embeddings in vector databases  
**Context Windows:**Max short-term memory, not ideal amount of information to process  
Chunking and data types are the two pieces of our data puzzle

**Metadata** data stored along with your vector embeddings  
 about data or the way it was processed  
 Metadata for RAG must include the text itself  
used to filter searches ex:   
 section title, paragraph number and chunk location  
 embedded data, publication date, and author  
  
**Chunking** How we make documents consumable for generative AI use like RAG.  
 No of chunks that fits is based on how you design your app.  
 Ensure that its possible to consume chunks in 1 go

Consumable, Coherent and Contextual

**Coherent:** Does it make Sense?Doesn’t stop mid word, clause or sentence? **Ex: “**Curiosity killed the cat**”** is Coherentbut“killed the”is not  
  
**Contextual:**Contains all the necessary context to answer a question  
  
**Considerations for Chunking:**Chunk Size-  
 How many characters at a time?  
 Paragraphs = 100 characters  
 Depends on Document Format  
 Character Overlap between chunks  
 Retains Context between Chunks  
   
Chunk Overlap  
Special CharactersSplit when you see the characterHelps with CoherenceExamples:”.”,”\n\n”,”.\n”

**Chunking Examples:**  
Document Data – Research paper/ Books  
Q/A Trascript - AMA that you find on REDITT, Podcasts, classes, talks, short long snippets, Chunks can be semantically linked  
Chat Transcript – Paper, Reports, Documentation  
 Paper – Research/essays/blogposts

Reports - works/news/lab reports  
Documents - Write-Ups, guides or API docs  
Regularly sized chunks of data  
split on double lines

Combine multiple QAs until limit  
Metadata of different chunks  
Customer support , texts, DMS  
Irregular Sized Chunks  
Special characters are important  
  
**Introduction to Embeddings**Turns your text into Vectors  
Deep Neural Networks.  
Turn unstructured data into embedding vectors.  
ResNet50, Sentence Transformers Family, Whisper.  
Hugging Face MTEB is a great source to find models.  
Different formats need different models  
  
**Embedding Considerations**

Picking the right model  
What to embed  
How to compare  
Embedding Size, Model Size, Training Data  
When you put into a neutral net.  
Must be same size to compare.  
only embeddings of the same size can be compared.  
smaller model = less expensive  
Large model = more fine grained  
Embedding model is not equal to LLM  
Data quality =model quality

**Algortithmetic Models**

TF-IDF Term frequency – Inverse Document Frequency.  
SPLADE – Sparse Lexical And expansion.  
BM-25 - Best Matching 25

**What to Embed**Sentence/Text itself  
Large to small  
small to large

Cosine  
Inner Product  
Euclidean

Basic – Just embed the Chunk, sometimes it works but requires some finesse  
Small to Big – Embed a sentence , store paragraph as text, good for context  
Big to Small – embed a paragraph, store a sentence, post processing Chunk LLM  
MTEB different models  
  
**MetaData**?

Non Embedding data  
Traditional DBS  
Advanced usage like filtering  
  
Chunking and Non Chunking  
-> sentence number, subtitle, section header  
->used for context and filtering

Non chunking   
->Meta data not tied to the chunking process.  
-> author, last updated date, document title  
-> used for filteting  
  
**Storing Metadata**:  
  
Can be linked.  
Easier to Store with Vectors.  
cite your sources.

**class langchain\_text\_splitters.character.CharacterTextSplitter(  
separator: str**='\n\n'**,is\_separator\_regex: bool**=False**,**\*\***kwargs: Any,)**this means if text goes over chunk size and does see any double new lines it may not form a new chunk . if we want to ensure to have correct chunk size, we should add a separator here own custom separator example new line

\n  
  
**but this is best practice**  
splitter = RecursiveCharacterTextSplitter(**separators=["\n\n", "\n", " ", ""]**, # fallback order chunk\_size=1000,  
 chunk\_overlap=100  
)

**Semantic Embeddings**

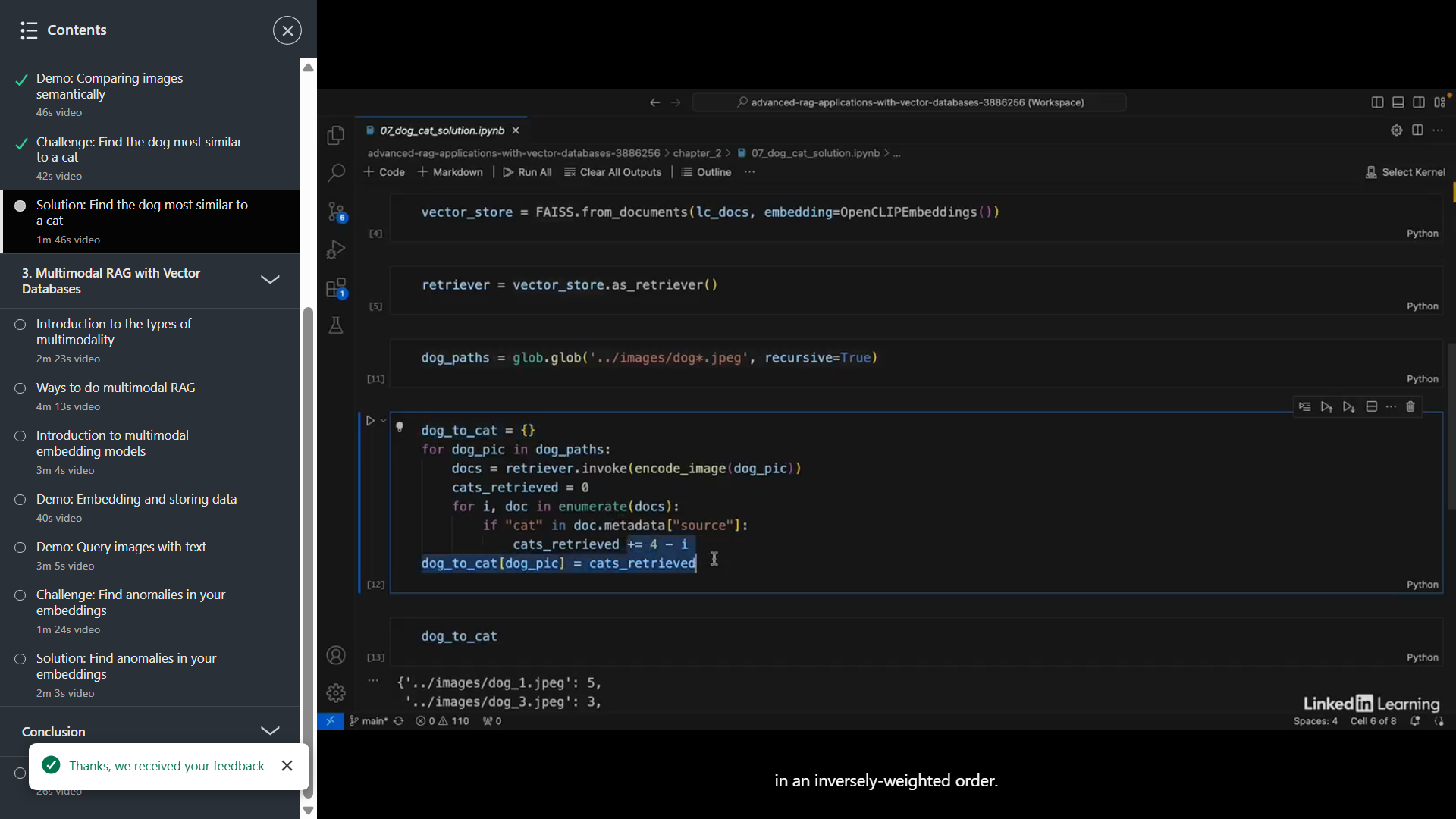
Derived from deep learning models.  
Image data gets passed from the input layer.  
Output of second to last layer of vision models.  
Captures what the image means.  
Remember for Rag we focus on the semantic embeddings.  
  
**Vision Models**:

Deep Neural Networks for Computer Vision.  
Classification of Images / segmentation or Object Detection.  
  
**History**:

1960: - deep, fully connected neural networks  
1993: - convolutional neural networks with max pooling  
2019:- vision transformers.  
 **Convolutional Neural Networks**:

Convolutional layer and pooling layer.  
Convolutions provide context from piece to piece by taking windows of images.  
Pooling combines Contexts  
 **Example**

2d IMAGE filled with numbers  
3\*3 convolution  
each convolution filter  
filter learnable add up resulting value to the square  
  
Vision transformers use encoder/decoder/attention  
Turn images into sets of patch embeddings  
N\*N squares  
Attention mechanism repeatedly transforms vectors of images



Multi Modality

Multi + Modal = different types  
  
Image+Text  
Image+Audio  
Video  
  
**Datatable examples**  
PDF+CSV  
Text+Table  
Table+Graph  
  
Ways to do MultiModal RAG  
One Multi modal embedding model  
Multiple Embedding Models and search Modes  
Models for different types of data route each type of data to the right modal for storing and searching

Multiple multimodal models

**Single multimodal model:**One vector store  
One Embedding Model  
Each embedding model produces will have same dimensionalityMultiple Models

**One Model for each data type  
One Vector store for each model  
Only Vectors of same size can be compared different training context  
Route vector stores**Multiple Multimodal model

Can re rank based on multiple embedding  
Different functionto embed each type of data