**Big data notes**

**Introduction to Data model**

**What is a Data Model​?**

Conceptual representation of the data structures that are required by a database.

The data structures include​

* the data objects​
* the associations between data objects​
* the rules which govern operations on the objects

**Different Data Models​**

* Conceptual: describes WHAT the system contains​
* Logical: describes HOW the system will be implemented, regardless​ of the DBMS​
* Physical: describes HOW the system will be implemented using a ​ specific DBMS

Elements of Data Models​

* Entity - A real world thing or an interaction between 2 or more real world things​
* Attribute - The atomic pieces of information that we need to know about entities​
* Relationship  - How entities depend on each other in terms of why the entities depend on each other and what that relationship is

Conceptual Data Model​

* The focus is to represent data as a user will see it in the "real world" ​
* The main aim of this model is to establish the entities, their attributes, and their relationships.

Logical Data Model​

* Logical data models add further information to the conceptual model elements.​
* Defines the structure of the data elements and set the relationships between them

Physical Data Model​

* It describes the database specific implementation of the data model.​
* It offers an abstraction of the database and helps generate schema.

Advantages of Data Models​

* ​To make sure that the data objects offered by the functional team are represented accurately.​
* Should be detailed enough to be used for building the physical database.​
* Information in the data model can be used for defining the relationship between tables, primary and foreign keys, and stored procedures.​
* It helps business to communicate within and across organizations.​
* It helps to recognize correct sources of data to populate the model.

Disadvantages

* To develop a Data model, one should know the physical characteristics of the data.​
* It requires knowledge of the hidden truth related to the system.​
* Even smaller change made in structure require modification in the entire application.​
* There is no set data manipulation language in DBMS.

Common data modelling notations​

1. Barker’s Notation​
2. Chen Notation​

**In Chen Notation, we distinguish types of entities:**

Entity - rectangle

Weak Entity

Associative Entity

**Attributes – Oval**

Key Attribute (underscored)

Partial Key Attribute (dashed underscored)

**Multivalued Attribute (ex: hobby)** -  Multivalued attribute is depicted by a dual oval

**Derived Attribute (dashed Oval)**

Some attributes can be further subdivided into smaller parts and are called**composite attributes**

**Strong relationship - single rhombus**

**Weak (identifying) relationship -**A relationship where Child entity is existence-dependent on parent, and PK of Child Entity contains PK component of Parent Entity​. This relationship is represented by a double rhombus

**Cardinality**

**Participation Constraints**

1. IDEF1X Notation​

**IDEF1X** (Integration DEFinition for Information Modeling) is a method for designing relational databases

1. Arrow Notation​
2. UML Notation​

**Generalization** is represented by an empty arrow​

**Aggregation** is shown as binary association with a hollow diamond as a symbol of the aggregation at the end of the association line​

**Composition** is presented as binary association with a black diamond as a symbol of the composition at the end of the association line

1. Crow’s Foot Notation​

​Also known as IE notation

* An entity is a representation of a class of object ​
* It can be a person, place, thing, etc​
* Entities usually have attributes that describe them​
* In crow’s foot notation, an entity is represented by a rectangle, with its name on the top​
* The name is singular (entity) rather than plural (entities)

**How to Model Data:​**

1. Identify entity types​
2. Identify attributes​
3. Apply naming conventions​
4. Identify relationships​
5. Apply data model patterns​
6. Assign keys​
7. Normalize to reduce data redundancy

**Elementary data analysis**

**Standard normal distribution**

Z Score enables you to compare observations with different means and standard deviations and place them on a standard scale.›This process is called standardization, and it allows you to compare observations and calculate probabilities across different populations.

To calculate the standard score for an observation, take the raw measurement, subtract the mean, and divide by the standard deviation.

Z=( x – mu)/sigma

**Shape of a Distribution**

First: If the data values seem to pile up into a single "mound", we say the distribution is unimodal.  
If there appear to be two "mounds", we say the distribution is bimodal.  
If there are more than two "mounds", we say the distribution is multimodal.  
Second: we focus on whether the distribution is symmetric, or if it has a longer "tail" on one side or another. In case where there is a longer "tail", we say the distribution is skewed in the direction of the longer tail.

**Bayesian Statistics**

It is defined as: Probability of an event Agiven Bequals the probability of Band Ahappening together divided by the probability of B.

**Apache Mahaout**

The components provided by Mahout to build a recommender engine are as follows:​

* *DataModel*​
* *UserSimilarity*​
* *ItemSimilarity*​
* *UserNeighborhood*​
* *Recommender*

**Steps to create a Recommender using Mahout​**

**Step1: Create DataModel Object​**

* The constructor of PearsonCorrelationSimilarity class requires a data model object.​
* File must have the Users, Items, and Preferences details of a product.

**Step2: Create UserSimilarity Object​**

* In this example, we want to create a user-based recommender. ​
* When we compute recommendations for a particular user, we look for other users with a similar taste and pick the recommendations from their items.​
* For finding similar users, we have to compare their interactions.​
* One popular method is to compute the [correlation coefficient](https://en.wikipedia.org/wiki/Pearson_product-moment_correlation_coefficient) between their interactions.​

**Step3: Create UserNeighborhood  Object​**

* The next thing we have to do is to define which similar users we want to leverage for the recommender.​
* There are two types of neighborhoods:​
* NearestNUserNeighborhood - This class computes a neighborhood consisting of the nearest *n* users to a given user. "Nearest" is defined by the given UserSimilarity.​
* ThresholdUserNeighborhood - This class computes a neighborhood consisting of all the users whose similarity to the given user meets or exceeds a certain threshold. Similarity is defined by the given UserSimilarity. For simplicity, here we use ThresholdUserNeighborhood

**Step4: Create Recommender  Object​**

* Now we have all the pieces to create our recommender.​
* Create UserbasedRecomender object and pass all the above created objects to its constructor as shown below.​
* UserBasedRecommender recommender = new GenericUserBasedRecommender(model, neighborhood, similarity); ​

**Step5: Recommend Items to a User​**

* Recommend products to a user using the recommend() method of Recommender interface.​
* This method requires two parameters. The first represents the user id of the user to whom we need to send the recommendations, and the second represents the number of recommendations to be sent.​

List<RecommendedItem> recommendations = recommender.recommend(2, 3); ​

    for (RecommendedItem recommendation : recommendations) ​

    { ​

System.out.println(recommendation);​

}

**Procedure of Clustering**​

**To cluster the given data you need to :​**

* Start the Hadoop server. ​
* Create required directories for storing files in Hadoop File System.​
* Create directories for input file, sequence file, and clustered output in case of canopy.​
* Copy the input file to the Hadoop File system from normal file system.​
* Prepare the sequence file from the input data.​
* Run any of the available clustering algorithms.​
* Get the clustered data.

The following steps are to be followed to implement Classification:​

* Generate example data​
* Create sequence files from data​
* Convert sequence files to vectors​
* Train the vectors​
* Test the vectors​

**Apache Hive**

* Hive is an open source data warehouse system built on top of Hadoop.​
* Facilitates reading, writing, and managing large datasets residing in HDFS.​
* It uses SQL for querying and analyzing large set of structured and semi structured data stored in cluster.​
* It is a design for OLAP which stores schema in a database and processed data in HDFS.​
* It provides SQL type language for querying called HiveQL or HQL which simplifies the vey tedious job os writing MapReduce programs.​
* HiveQL automatically translates SQL-like queries into MapReduce jobs

**1. Metastore​**

* It stores metadata for each of the tables like their schema and location.​
* Hive also includes the partition metadata which helps the driver to track the progress of various data sets distributed over the cluster.​
* It stores the data in a traditional RDBMS format.​

**2. Driver​**

* It acts like a controller which receives the HiveQL statements. ​
* The driver starts the execution of the statement by creating sessions.
* It monitors the life cycle and progress of the execution.​
* Driver stores the necessary metadata generated during the execution of a HiveQL statement.​
* Acts as a collection point of data or query result obtained after the Reduce operation.​

**3. Compiler**​

* It performs the compilation of the HiveQL query which converts the query to an execution plan.
* The plan contains the tasks and steps needed to be performed by the MapReduce to get the output as translated by the query.​
* The compiler in Hive converts the query to an **Abstract Syntax Tree.**​
* First, check for compatibility and compile-time errors, then converts the AST to a **Directed Acyclic Graph.**​

**4. Optimizer**​

* It performs various transformations on the execution plan to provide optimized DAG.
* It aggregates the transformations together.​
* It can also split the tasks, such as applying a transformation on data before a reduce operation, to provide better performance.​

**5. Executor**​

* Once compilation and optimization complete, the executor executes the tasks.​
* Executor takes care of pipelining the tasks.​
* It also provides a Web UI, CLI and HDInsight(MS) for users to interact.
* CREATE TABLE IF NOT EXISTS employee ( eid int, name String, salary String, destination String) COMMENT ‘Employee details’ ​
* ROW FORMAT DELIMITED ​
* FIELDS TERMINATED BY ‘\t’ ​
* LINES TERMINATED BY ‘\n’ ​
* STORED AS TEXTFILE;

**AWS quicksight**

**Corelation Pattern**

**Bayesian staistics**