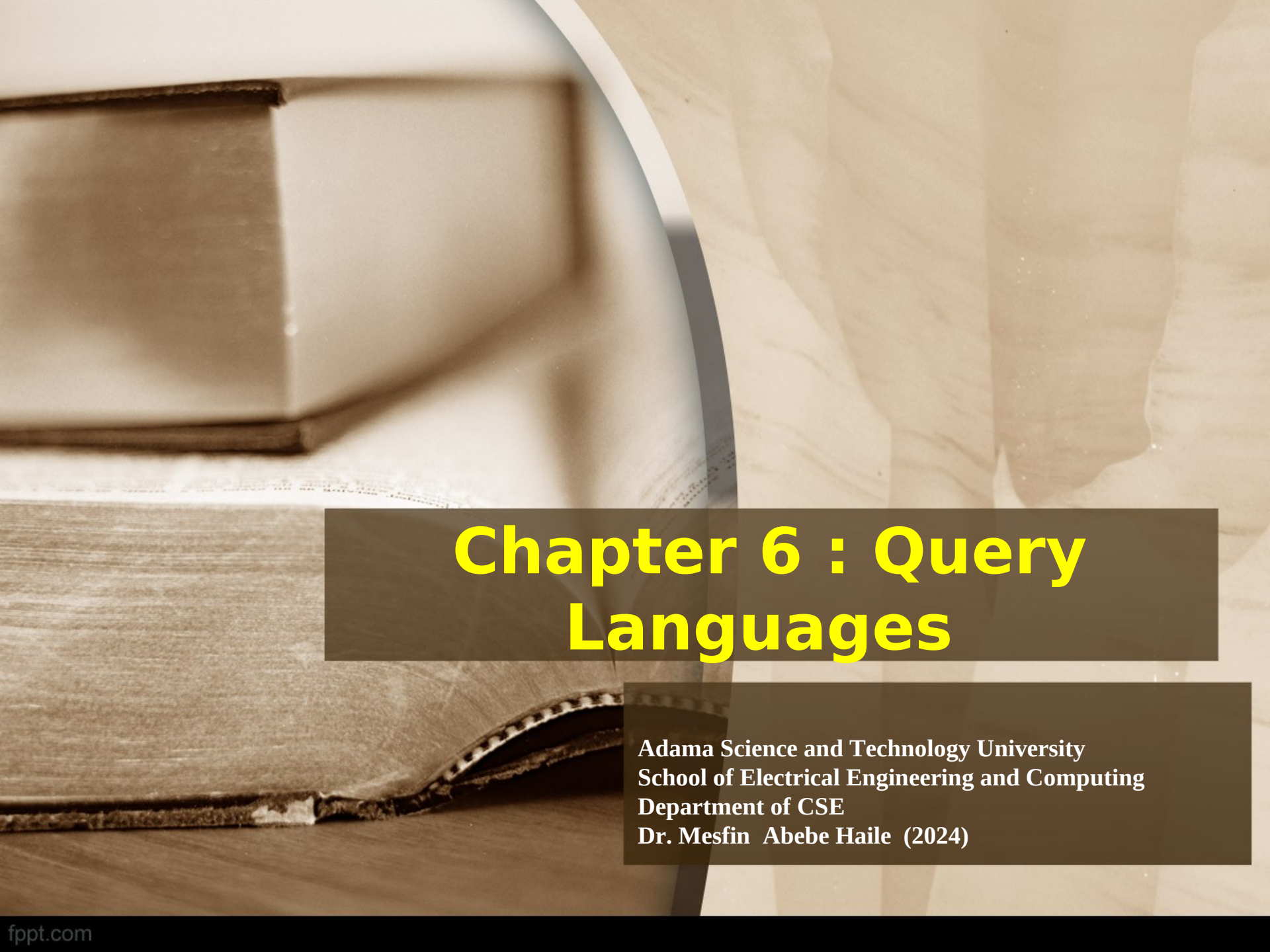




Chapter 6 : Query Languages

Adama Science and Technology University
School of Electrical Engineering and Computing
Department of CSE
Dr. Mesfin Abebe Haile (2024)

Keyword-based Querying

- ❖ Queries are *combinations of words*.
- ❖ The document collection is searched for documents that *contain these words*.
- ❖ Word queries are *intuitive, easy* to express and provide *fast ranking*.
- ❖ The *concept of word* must be defined:
 - ❖ A word is a *sequence of letters* terminated by a separator (period, comma, space, etc).
 - ❖ Definition of *letter* and *separator is flexible*; e.g., hyphen could be defined as a letter or as a separator.
 - ❖ Usually, *common words* (such as “a”, “the”, “of”, ...) are ignored.

Single-word Queries

- ❖ A query is a *single word*.
 - ❖ Usually used for *searching* in document images.
- ❖ *Simplest* form of query.
- ❖ What are the possible documents *retrieved as relevant*?
 - ❖ All documents that *include this word* are retrieved.
- ❖ On what base *documents are ranked*?
 - ❖ Documents may be *ranked by the frequency* of the *query word* in the document.
 - ❖ Documents containing *more of the query word* are given the *highest priority*.

Phrase Queries

- ❖ A query is a *sequence of words* treated as a *single unit*. Also called “*literal string*” or “*exact phrase*” query.
 - ❖ Phrase is usually surrounded by *quotation marks*.
 - ❖ All documents that include this *phrase are retrieved*.
- ❖ Usually, separators (commas, colons, ...) & common words (“a”, “the”, “of”, “for”...) in the *phrase are ignored*.
- ❖ In effect, this query is for a *set of words* that must *appear in sequence*.
 - ❖ Allows users to *specify a context* and thus *gain precision*.
 - ❖ Ex.: “Information Processing for Document Retrieval”.
- ❖ What are the possible *documents retrieved* as *relevant*?
 - ❖ All documents that include *phrase query* are retrieved.
- ❖ On what base *documents are ranked*?

Multiple-word Queries

- ❖ A query is a *set of words* (or phrases).
 - ❖ Ex.: What is the result for the query “*Data Mining and Intelligent Database Design*”?
- ❖ What are the possible documents *retrieved as relevant*?
 - ❖ *Two options*: A document is retrieved if it includes:
 - ❖ Any of the query words, or
 - ❖ each of the query words.

Multiple-word Queries

- ❖ On what bases documents be *ranked to list* according to *best matching principle*?
 - ❖ Documents are ranked by the number of *query words* they contain.
 - ❖ A document containing n query words is ranked higher than a document containing $m < n$ *query words*.
- ❖ Documents are ranked in *decreasing order*:
 - ❖ Those containing all the *query words are ranked* at the top, *only one query word at bottom*.
- ❖ *Frequency counts* may be used to break ties among documents that contain the *same query words*.

Boolean Queries

- ❖ Queries are formulated based *on concepts from logic: AND, OR, NOT*.
 - ❖ It describes the *information needed* by relating multiple words with *Boolean operators*.
- ❖ **Semantics**: For each query word w a corresponding set D_w is constructed that includes the *documents that contain w* .
- ❖ The *Boolean expression* is then interpreted as an expression on the corresponding document sets with *corresponding set operators*:
 - ❖ **AND**: Finds only documents containing *all of the specified words or phrases*.
 - ❖ **OR**: Finds documents containing *at least one of the specified words or phrases*.
 - ❖ **NOT**: Excludes documents containing the *specified word or phrase*.

Examples: Boolean Queries

- ❖ 1. Computer OR server
 - ❖ Finds documents containing *either computer, server or both*.
- ❖ 2. (computer OR server) NOT mainframe
 - ❖ Select all documents that discuss *computers or servers*, do not select any documents that *discuss mainframes*.
- ❖ 3. Computer NOT (server OR mainframe)
 - ❖ Select *all documents that discuss computers*, and do not discuss either servers or mainframes.
- ❖ 4. Computer OR server NOT mainframe
 - ❖ Select all documents that discuss *computers*, or documents that discuss *servers* but *do not discuss mainframes*.

Weighted Queries

- ❖ Each of the words is assigned a *different weight*, expressing the *relative importance* of the word within the query.
- ❖ A query is then a *set of word-weight pairs*:
 $(q_1, w_1), (q_2, w_2), \dots, (q_n, w_n)$.
- ❖ The *ranking* of a document is the *sum of the weights* for the query words that it satisfies.
- ❖ **Example**: given Query: (A,0.8), (B,0.9), (C,0.3); and
- ❖ **Document 1**: (A, B, D) and **Document 2**: (A, C, D) which document *ranked first* ?
 - ❖ Score of Document 1: $0.8+0.9 = 1.7$
 - ❖ Score of Document 2: $0.8+0.3 = 1.1$
 - ❖ Each document includes *two words from the query*, but Document 1 is ranked higher because it includes *more important words*.

Penalizing Documents

- ❖ When interpreting *queries*,
 - ❖ The Boolean model does not “penalize” documents with *extra* (non-requested) *keywords*.
 - ❖ Some models demote documents that include *keywords* that were *not requested*.
 - ❖ The *vector model* with the cosine measure
 - ❖ The *probabilistic* Bayesian network model
- ❖ Ex.: Assume the vector model with the *cosine measure* and the *simple case* that both documents and queries use *binary values*.
 - ❖ Consider the following *two documents* and a *query*:
 - ❖ $d_1 = (0,1,0,1,0)$, $d_2 = (0,1,1,1,0)$, $q = (0,1,0,1,0)$
 - ❖ $\text{sim}(q, d_1) = 1.0$, $\text{sim}(q, d_2) = 0.82$
 - ❖ d_2 is demoted because it includes an *extra keyword not requested* by q .

Pattern Queries

- ❖ What is Pattern?
 - ❖ An expression that defines a *set of objects*. Pattern shows the *internal representation of an object*.
 - ❖ What is the *pattern of a word*?
- ❖ *Pattern matching*: A word matches a pattern if it is equal to one of the words defined by *the pattern*.
- ❖ In other words,
 - ❖ The semantics are of *disjunction*: A pattern P that defines a *word* $(c_1, c_2, \dots, c_n)$ is *interpreted* as $c_1 \vee c_2 \vee \dots \vee c_n$.

Pattern Queries

- ❖ ***Similarity pattern***. Specifies a string and a radius
 - ❖ Defines all the words whose *distance from* the string is within the radius.
 - ❖ Assume the distance between two strings is measured by the number of *one-character changes* (insertions, deletions, replacements) required to transform one string into the other.
 - ❖ The *similarity pattern* (*king*, 2) defines *kin*, *kong*, *knig*, *kings*, *cling*, ...
- ❖ Useful to compensate for *typing or scanning* (OCR) errors.
 - ❖ One of the technique used for pattern matching is *string editing*.

String Editing

- ❖ The problem is given *two sequences of symbols*, $X = x_1 x_2 \dots x_n$ and $Y = y_1 y_2 \dots y_m$, transform X to Y , based on a sequence of three operations: *Delete*, *Insert* and *Replace*, so that for every operation $\text{COST}(C_{ij})$ is incurred.
- ❖ The objective of string editing is to identify a *minimum cost sequence* of edit operation that will transform X into Y .
- ❖ Example: consider the *sequences*:
 - ❖ $X = \{a a b a b\}$ and $Y = \{b a b b\}$
- ❖ Identify a *minimum cost sequence* of edit operation that transform X into Y .
- ❖ Assume change *costs 2 units*, delete *1 unit* and insert *1 unit*.

Dynamic programming

- ❖ The minimum cost of any *edit sequence that transforms* $x_1 x_2 \dots x_i$ into $y_1 y_2 \dots y_j$ (for $i > 0$ and $j > 0$) is the minimum of the three costs: *delete*, *replace*, or *insert operations*.
- ❖ The following *recurrence equation* is used for $\text{COST}(i,j)$.

$$\begin{array}{llll}
 0 & \text{if } i=0, j=0 & & \\
 \text{COST}(i-1,0) + D(x_i) & i > 0, j=0 & \text{COST}(0,j-1) + I(y_j) & j > 0, i=0 \\
 \text{COST}'(i,j) & i > 0, j > 0 & & \\
 \text{where } \text{COST}'(i,j) = \min \{ \text{COST}(i-1,j) + D(x_i), & \text{COST}(i-1,j-1) + C(x_i,y_j), & \text{COST}(i,j-1) + I(y_j) & \\
 \} & & &
 \end{array}$$

$$\text{COST}(i,j) = \left\{ \begin{array}{l} 0 \quad \text{if } i=0, j=0 \\ \text{COST}(i-1,0) + D(x_i) \quad i > 0, j=0 \\ \text{COST}(0,j-1) + I(y_j) \quad j > 0, i=0 \\ \min \{ \text{COST}(i-1,j) + D(x_i), \text{COST}(i-1,j-1) + C(x_i,y_j), \text{COST}(i,j-1) + I(y_j) \} \quad i > 0, j > 0 \end{array} \right.$$

Example

❖ Transform the *sequences*:

❖ $X_i = \{a \ a \ b \ a \ b\}$ into $Y_j = \{b \ a \ b \ b\}$

❖ With *minimum cost sequence* of edit operation using dynamic programming approach, Assume that *change costs 2 units*, *delete* and *insert* 1 unit.

j \ i	0	1	2	3	4
0	0	1	2	3	4
1	1	2	1	2	3
2	2	3	2	3	4
3	3	2	3	2	3
4	4	3	2	3	4
5	5	4	3	2	3

- ❖ The value 3 at (5,4) is the optimal solution
- ❖ By tracing back one can determine which operations lead to *optimal solution*.
- ❖ Delete x_1 , Delete x_2 and Insert y_4 Or,
- ❖ Change x_1 to y_1 & Delete x_4 .

Natural language

- ❖ Using natural language for querying is *very attractive*.
- ❖ *Example*: Find all the documents that discuss
 - ❖ “ campaign finance reforms, including documents that discuss violations of campaign financing regulations.
 - ❖ Do not include documents that discuss campaign contributions by the gun and the tobacco industries”.
- ❖ Natural language queries are converted to a formal language for processing against a *set of documents*.
- ❖ Such translation requires *intelligence* and is still a *challenge*.

Natural language

- ❖ Pseudo NL processing: System scans the text and extracts recognized terms and *Boolean connectors*.
- ❖ The *grammaticality of the text* is not important.
 - ❖ Often used by *search engines*.
- ❖ **Problem**: Recognizing the negation in the *search statement* (“Do not include...”).
- ❖ **Compromise**: Users enter natural language clauses connected with *Boolean operators*.
- ❖ In the above example: “*campaign finance reforms*” or “*violations of campaign financing regulations*” and not “*campaign contributions by the gun and the tobacco industries*”.

Question & Answer



Thank You !!!