Database Systems
CSE 514

Lecture 8: Data Cleaning and Sampling
Announcements

• WQ7 was due last night (did you remember?)

• HW6 is due on Sunday
  – Weston will go over it in the section

• The final exam is next Tuesday, in class
Data Cleaning

• Cover slides 2-20 from the Data Cleaning Tutorial
Deduplication

• Other names: entity resolution, record linkage, ...

• Cover (quickly!) slides 23-38 of the ER tutorial
Record Linkage

• The same object is represented in different ways in two relations
• Problem: find matching pairs

• Examples
  – Different naming conventions: “IBM” v.s. “International Business Machines Corporation”
## Example

<table>
<thead>
<tr>
<th>Company1</th>
<th>Company1</th>
</tr>
</thead>
<tbody>
<tr>
<td>CName1</td>
<td>. . . Other attributes</td>
</tr>
<tr>
<td>Microsoft Corp</td>
<td></td>
</tr>
<tr>
<td>Apple Computer</td>
<td></td>
</tr>
<tr>
<td>Apples, Pears, and More</td>
<td></td>
</tr>
<tr>
<td>. . .</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>CName2</th>
<th>. . . Other attributes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Microsoft Inc</td>
<td></td>
</tr>
<tr>
<td>Apple Corporation</td>
<td></td>
</tr>
<tr>
<td>Apples and Pears Farm</td>
<td></td>
</tr>
<tr>
<td>. . .</td>
<td></td>
</tr>
</tbody>
</table>
Fellegi & Sunter Model

• Each relation has several attributes:
  \( R(A_1,A_2,...,A_k) \)
  \( S(A_1,A_2,...,A_k) \)

• Fix two records \( r \) and \( s \): we have a score of similarity for each pair of attributes:
  \( \text{sim}(r.A_1,s.A_1) \),
  \( \text{sim}(r.A_2,s.A_2) \), ..., 
  \( \text{sim}(r.A_k,s.A_k) \)

• Using these scores, decide if \( r = s \) or \( r \neq s \)

• This lecture will discuss only single attribute
Similarity Join

another name for Fuzzy Matching

```
SELECT *
FROM Company1, Company2
WHERE cname1 ≈ cname2
```

Intended meaning: retrieve all pairs that are “sufficiently similar”
Outline for Today

• Define similarity $s_1 \approx s_2$
  – Jaccard, Hamming, Edit distance

• Given a set of sets $s_1, s_2, \ldots$ compute efficiently all pairs $(s_i, s_j)$ s.t. $s_i \approx s_j$
  – Min-hashes
  – Locality Sensitive Hashing
Similarity Functions
What is “Similar”?

• Similarity function $\text{sim}(s_1, s_2)$:
  – $\text{Sim}(s_1, s_2) > k$ means $s_1, s_2$ are similar

• Distance function $\text{dist}(s_1, s_2)$:
  – $\text{Dist}(s_1, s_2) < k$ means $s_1, s_2$ are similar
Two Approaches

• Q-grams
  – Simple, and can lead to efficient optimizations

• Edit Distance
  – More accurate, but less amenable to optimizations
Q-Grams

• Given a string s, a q-gram is a substring of length q
• Usually q = 3

\[
\text{washington} \quad \text{woshington}
\]

\[
s1 = \{\text{was, ash, shi, hin, ing, ngt, gto, ton}\}
\]

\[
s2 = \{\text{wos, osh, shi, hin, ing, ngt, gto, ton}\}
\]

Variation: may include beginning and end: ##w, #wa, on$, n$$
Hamming Distance

- $H(s_1, s_2) = |s_1 \Delta s_2| = |s_1 - s_2| + |s_2 - s_1|

“$s_1$ is similar to $s_2$” if $H(s_1, s_2) < c$, for some constant $c$
Hamming Distance

s1 = \{was, ash, shi, hin, ing, ngt, gto, ton\}

s2 = \{wos, osh, shi, hin, ing, ngt, gto, ton\}

H(s1, s2) = ?
Hamming Distance

\[ s1 = \{ \text{was, ash, shi, hin, ing, ngt, gto, ton} \} \]
\[ s2 = \{ \text{wos, osh, shi, hin, ing, ngt, gto, ton} \} \]

\[ H(s1, s2) = 4 \]
Jaccard Similarity

- $J(s_1, s_2) = \frac{|s_1 \cap s_2|}{|s_1 \cup s_2|}$

Note: $J(s_1, s_2) \in [0, 1]$

“s1 is similar to s2” if $J(s_1, s_2) > k$, for some constant k
Jaccard Similarity

\[ s1 = \{\text{was, ash, shi, hin, ing, ngt, gto, ton}\} \]
\[ s2 = \{\text{wos, osh, shi, hin, ing, ngt, gto, ton}\} \]

\[ J(s1, s2) = ? \]
Jaccard Similarity

\[ J(s_1, s_2) = \frac{6}{10} = 0.6 \]
They are related!

• Suppose $|s_1| = |s_2| = L$

• $|s_1 \cap s_2| = L - H(s_1, s_2)$

• $|s_1 \cup s_2| = L + H(s_1, s_2)$

• $J(s_1, s_2) = (L - H(s_1, s_2)) / (L + H(s_1, s_2))$
Representing q-Grams

Company (id, name, …) CQ(id, qgram)

<table>
<thead>
<tr>
<th>Id</th>
<th>Name</th>
<th>…</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Washington</td>
<td>…</td>
</tr>
<tr>
<td>2</td>
<td>…</td>
<td>…</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Id</th>
<th>Qgram</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>was</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>ash</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>shi</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>hin</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>…</td>
<td></td>
</tr>
</tbody>
</table>

The q-gram schema
Naive Similarity Joins

SELECT *
FROM Company1, Company2
WHERE |cname1 ∩ cname2| > 6

Company1(id1, cname1, …)
Company2(id2, cname2, …)

CQ1(id1, qgram1)
CQ2(id2, qgram2)

Rewrite the query over the q-gram schema
Naive Similarity Joins

```sql
SELECT * , y.*
FROM Company1, Company2
WHERE |cname1 \cap cname2| > 6
```

```sql
SELECT x.*, y.*
FROM Company1 x, Company2 y, CQ1 u, CQ2 v
WHERE x.id = u.id and y.id = v.id
  and u.qgram = v.qgram
GROUP BY x.id, y.id
HAVING count(*) > 6
```
Edit Distance

- Sometimes none of the similarity/distance measures are good enough
- Need to use edit distance
- Models more accurately the typing mistakes
- Slower than Jaccard or Hamming
- Use Jaccard or Hamming for pruning, then compute edit distance to remove false positives
The Levenstein Distance

- Distance is **shortest sequence of edit commands** that transform $s$ to $t$.
- Simplest set of operations:
  - Copy character from $s$ over to $t$
  - Delete a character in $s$ (cost 1)
  - Insert a character in $t$ (cost 1)
  - Substitute one character for another (cost 1)
Levenstein distance - example

- distance(“William Cohen”, “Willliam Cohon”)

\[
s_1 \quad \text{W I L L I A M \_ C O H E N}
\]

\[
s_2 \quad C \ C \ C \ C \ I \ C \ C \ C \ C \ C \ C \ C \ C \ C \ C \ s \ C
\]

\[
op \quad 0 \ 0 \ 0 \ 0 \ 0 \ 1 \ 1 \ 1 \ 1 \ 1 \ 1 \ 1 \ 1 \ 1 \ 1 \ 2 \ 2 \ 2
\]

\[
\text{cost}
\]
Levenstein distance - example

- distance("William Cohen", "Willlliam Cohon")

\[
\begin{array}{cccccccccccccccc}
  & W & I & L & L & \text{gap} & I & A & M & _ & C & O & H & E & N \\
s1 & \_ & \_ & \_ & \_ & \_ & \_ & \_ & \_ & \_ & \_ & \_ & \_ & \_ & \_ & \_ \\
  & W & I & L & L & L & I & A & M & _ & C & O & H & O & N \\
op & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 2 & 2 \\
\end{array}
\]

alignment

cost
Computing Levenshtein distance 1/4

\[ D(i, j) = \text{score of best alignment from } s_{1..i} \text{ to } t_{1..j} \]

= min \[
\begin{cases} 
D(i-1, j-1), & \text{if } s_i = t_j \quad \text{//copy} \\
D(i-1, j-1)+1, & \text{if } s_i \neq t_j \quad \text{//substitute} \\
D(i-1, j)+1 & \quad \text{//insert} \\
D(i, j-1)+1 & \quad \text{//delete}
\end{cases}
\]
Computing Levenshtein distance 2/4

\[ D(i,j) = \text{score of best alignment from } s1..si \text{ to } t1..tj \]

\[ = \min \left\{ \begin{array}{l} D(i-1,j-1) + d(si,tj) \quad \text{//subst/copy} \\ D(i-1,j)+1 \quad \text{//insert} \\ D(i,j-1)+1 \quad \text{//delete} \end{array} \right\} \]

(simplify by letting \( d(c,d) = 0 \) if \( c=d \), 1 else)

also let \( D(i,0)=i \) (for \( i \) inserts) and \( D(0,j)=j \)
Computing Levenshtein distance $3/4$

$$D(i,j) = \min \begin{cases} 
    D(i-1,j-1) + d(s_i,t_j) & \text{//subst/copy} \\
    D(i-1,j) + 1 & \text{//insert} \\
    D(i,j-1) + 1 & \text{//delete}
\end{cases}$$

\[
\begin{array}{cccccc}
 & C & O & H & E & N \\
M & 1 & 2 & 3 & 4 & 5 \\
C & 1 & 2 & 3 & 4 & 5 \\
C & 2 & 2 & 3 & 4 & 5 \\
O & 3 & 2 & 3 & 4 & 5 \\
H & 4 & 3 & 2 & 3 & 4 \\
N & 5 & 4 & 3 & 3 & 3 \\
\end{array}
\]

$= D(s,t)$
Computing Levenstein distance 4/4

\[
D(i,j) = \min \begin{cases} 
D(i-1,j-1) + d(s_i, t_j) & \text{// subst/copy} \\
D(i-1,j) + 1 & \text{// insert} \\
D(i,j-1) + 1 & \text{// delete}
\end{cases}
\]

A trace indicates where the min value came from, and can be used to find edit operations and/or a best alignment (may be more than 1).
Needleman-Wunch distance

\[ D(i,j) = \min \begin{cases} 
D(i-1,j-1) + d(s_i,t_j) & /\text{subst/copy} \\
D(i-1,j) + G & /\text{insert} \\
D(i,j-1) + G & /\text{delete} 
\end{cases} \]

\[ d(c,d) \text{ is an arbitrary distance function on } \]
\[ \text{characters (e.g. related to typo frequencies, amino acid substitutability, etc)} \]

\[ G = \text{“gap cost”} \]
Pre-filtering

• If editDistance(s1, s2) ≤ k,
  then |Qgrams(s1) ∩ Qgrams(s2)| ≥ v

• Where v = max(|s1|,|s2|) – 1 – (k-1) * q

• Where q = length of the q-gram

• Thus, can use any of the signature schemes
Jaccard

• Recap:
  Jaccard similarity:  \( J(S, T) = \frac{|S \cap T|}{|S \cup T|} \)

• Problem: given large collection of sets \( S_1, S_2, \ldots, S_n \), and given a threshold \( s \), find all pairs \( S_i, S_j \) s.t. \( J(S_i, S_j) > s \)
Similarity Joins
Similarity joins

Problem:
- We are given $n$ sets $s_1, s_2, ..., s_n$
- Find quickly all pairs s.t. $J(S_i, S_j) > s$
- False positives, false negatives are OK
Application 1: Collaborative Filtering

- We have $n$ customers: 1, 2, ..., $n$
- Each customer $i$ buys a set of items $S_i$
- We would like to recommend items bought by customer $j$, if $J(S_i, S_j) > s$
Application 2: Similar Documents

• Given n documents $d_1, d_2, \ldots, d_n$
• Let $S_i$ be the set of q-grams for document $i$
• Want to find all pairs of “similar” documents, i.e. for which $J(S_i, S_j) > s$
Example

• You work for a copyright violation detection company
• Customer: has $10^6$ documents
• Web: has $10^{11}$ pages

• Your job is to find “almost identical documents”
The Signature Method

• For each $S_i$, compute a signature $\text{Sig}(S_i)$ s.t.

\[
J(S_i, S_j) > s \iff \text{Sig}(S_i) \cap \text{Sig}(S_j) \neq \emptyset
\]

With high probability
The Signature Method

• Step 1: compute all pairs $i,j$ for which $\text{Sig}(S_i) \cap \text{Sig}(S_j) \neq \text{emptyset}$
  – This is a join operation!

• Step 2: for all such pairs, return $(i,j)$ if $J(S_i,S_j) > s$
  – Hopefully only a few such pairs

Both false positives and false negatives are possible
Discussion

- We will assume that each $S_i$ is a string, and we associate it with the set of its $q$-grams:

- The signature $\text{Sig}(S_i)$ is also a set of “something” (to be explained)
Signature 1 = the q-grams!

- Obviously: \(|s_1 \cap s_2| > k \Rightarrow s_1 \cap s_2 \neq \emptyset|

SELECT DISTINCT x.*, y.*
FROM Company1 x, Company2 y, CQ1 u, CQ2 v
WHERE x.id = u.id and y.id = v.id
    and u.qgram = v.qgram

Lots of false positives
Signature 2: Prefix Filter

• Suppose all sets $s_i$ have same cardinality $L$

• Order the q-grams lexicographically

• Define $\text{sig}(s) = \{\text{the first } L-k+1 \text{ smallest q-grams in } q\}$
Signature 2: Prefix Filter

- Example: $L = 8$, $k = 6$, $L - k + 1 = 3$

\[
s_1 = \{\text{was, ash, shi, hin, ing, ngt, gto, ton}\} = \{\text{ash, gto, hin, ing, ngt, shi, ton, was}\}
\]
\[
\text{Sig}(s_1) = \{\text{ash, gto, hin}\}
\]
Signature 2: Prefix Filter

• Fact: if $|s_1 \cap s_2| > k$
  then $\text{sig}(s_1) \cap \text{sig}(s_2) \neq \text{emptyset}$

• Why?
Signature 3: Minhash

• Let $\pi$ be an arbitrary permutation of the domain

• For each $i$, let:

$$mh(S_i) = \{\text{the smallest element in } S_i \text{ according to } \pi\}$$
Example

- The entire domain is \{a,b,c,d,e,f,g,h\}
- The set \(S_i\) is \(S_i = \{a,b,c,e,f\}\)
- Suppose we choose the permutation: \(\pi = d,g,c,h,b,f,a,e\)

Then what is \(mh(S_i) = ?\)
Minhash

Main property:

\[ \text{Probability}(mh(S_i) = mh(S_j)) = J(S_i, S_j) \]
Warmup Question

• Choose a *random* permutation $\pi$ of 
  \{a, b, c, ..., z\}

$S = \begin{array}{ccc}
  & b & d \\
  a & & c \\
\end{array}$

What is the probability that $mh(S) = c$?
Choose a random permutation $\pi$ of \{a,b,c...,z\}

$S =$

What is the probability that $mh(S) = c$ ?

Answer: $P = \frac{1}{4}$ (each of a,b,c,d can be the min)
What is $\text{Prob}(\text{mh}(S_i) = \text{mh}(S_j))$ ?
What is $\operatorname{Prob}(\operatorname{mh}(\text{Si}) = \operatorname{mh}(\text{Sj}))$?

Answer: $\frac{2}{6} = J(\text{Si},\text{Sj})$
Computing Minhashes

• We use a hash function (which we assume is random)

\[
\text{mh}(S) \{
    v = \infty;
    \text{forall } x \text{ in } S \text{ do }
    \quad \text{if } h(x) < v \text{ then } \{v = h(x); y = x;\}
    \text{return } y;
\}
\]
Example

• The set $S_i$ is $S_i = \{a, b, c, e, f\}$

• Compute $h$:
  $h(a) = 77, h(b) = 55, h(c) = 33, h(e) = 88, h(f) = 66$

• Then what is $mh(S_i) = \ ?$
Using Minhashes for Fuzzy Join

• Recall: we have n sets S1, ..., Sn
• Want to return all pairs s.t. \( J(S_i, S_j) > s \)

• Compute \( mh(S_1), ..., mh(S_n) \)
• Return pairs for which \( mh(S_i) = mh(S_j) \)

• The probability of returning \((S_i, S_j)\) is \( J(S_i, S_j) \)
• Probability of a false positive = \( J(S_i, S_j) \)
• Probability of a false negative = 1 - \( J(S_i, S_j) \)
Example

- We have \( n = 1,000,000 \) records \( S_1, \ldots, S_n \)
- We want pairs s.t. \( J(S_i, S_j) > 0.99 \)
- There are \( 10^{12} \) ordered pairs \((S_i, S_j)\)
- Suppose \( 2,000,000 \) pairs have \( J(S_i, S_j) > 0.99 \)
  the others have \( J(S_i, S_j) = 0.5 \)
- \# false positives = \( 0.5 \times 10^{12} \)
- \# false negatives = \( 0.01 \times 2,000,000 = 20,000 \)

This is bad
Naïve Idea

- Use $r$ independent hash functions $h_1, \ldots, h_r$
- For each $S_i$, compute a block of minhashes: $MH(S_i) = \text{the } r \text{ minhashes for each } j=1,\ldots,r$
- Return all pairs where $MH(S_i)=MH(S_j)$

- The probability of returning $(S_i,S_j)$ is $(J(S_i,S_j))^r$
- Probability of a false positive = $(J(S_i,S_j))^r$
- Probability of a false negative = $1-(J(S_i,S_j))^r$
Example

- \( r = 2 \)
- The set \( S_i \) is \( S_i = \{a, b, c, e, f\} \)
- Compute \( h_1 \):
  \[ h_1(a) = 77, h_1(b) = 55, h_1(c) = 33, h_1(e) = 88, h_1(f) = 66 \]
- Compute \( h_2 \):
  \[ h_2(a) = 22, h_2(b) = 66, h_2(c) = 55, h_2(e) = 11, h_2(f) = 44 \]
- Then what is \( MH(S_i) = ? \)
Example

- \( r = 2 \)
- The set \( S_i \) is \( S_i = \{a, b, c, e, f\} \)
- Compute \( h_1 \):
  \( h_1(a) = 77, h_1(b) = 55, h_1(c) = 33, h_1(e) = 88, h_1(f) = 66 \)
- Compute \( h_2 \):
  \( h_2(a) = 22, h_2(b) = 66, h_2(c) = 55, h_2(e) = 11, h_2(f) = 44 \)

- Then what is \( MH(S_i) = ? \)
- Answer: \( MH(S_i) = \text{ce} \) (an ordered pair)
Signature 4: LSH

Locality Sensitive Hashing

• b blocks of minhashes $\text{MH}_1, \ldots, \text{MH}_b$
• Each $\text{MH}_k$ has size $r$
• Total size of the signature: $m = b \times r$

• Return pairs $(S_i, S_j)$ s.t. there exists $k$ such that $\text{MH}_k(S_i) = \text{MH}_k(S_j)$
• Still a single simple join!
Each rectangle (=band) becomes one new hash value
Analysis

• Goal: want to compute the probability that $\text{Sig}(\text{MH}(S_i)) \cap \text{Sig}(\text{MH}(S_j)) \neq \emptyset$, as a function of $s = J(S_i, S_j)$

• Let $s = J(S_i, S_j)$
Analysis

What is the probability that two entries are equal?

\[ J(S_i,S_j) = s \]
J(S_i,S_j) = s

What is the probability that two entries are equal?

Answer: s
What is the probability that two entries are equal?

![Table and diagram](image)
J(Si, Sj) = s

What is the probability that two entries are equal?

Answer: $s^r$
J(S_i, S_j) = s

What is the probability that some pair of bands are equal?
$J(S_i, S_j) = s$

What is the probability that some pair of bands are equal?

Answer: $1-(1-s^r)^b$
This is precisely the probability that $\text{Sig}(S_i) \cap \text{Sig}(S_j) \neq \emptyset$

Answer: $1 - (1 - s^r)^b$
Analysis

P(s) = 1 - (1 - s^r)^b
Putting it together

• You work for a copyright violation detection company
• Customers: have documents 1, 2, 3, …., $10^6$
• Web: has pages 1, 2, 3, …., $10^{11}$

• Your job is to find “almost identical documents”
Step 1: Q-grams

**Doc**

<table>
<thead>
<tr>
<th>DocID</th>
<th>Qgram</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>To</td>
</tr>
<tr>
<td>1</td>
<td>o b</td>
</tr>
<tr>
<td>1</td>
<td>be</td>
</tr>
<tr>
<td>1</td>
<td>be</td>
</tr>
<tr>
<td>1</td>
<td>e o</td>
</tr>
<tr>
<td>1</td>
<td>or</td>
</tr>
<tr>
<td>1</td>
<td>or</td>
</tr>
<tr>
<td>1</td>
<td>r n</td>
</tr>
<tr>
<td>1</td>
<td>no</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>2</td>
<td>..</td>
</tr>
<tr>
<td>...</td>
<td></td>
</tr>
</tbody>
</table>

**Web**

<table>
<thead>
<tr>
<th>URL</th>
<th>Qgram</th>
</tr>
</thead>
<tbody>
<tr>
<td>abc.com</td>
<td>Oba</td>
</tr>
<tr>
<td>abc.com</td>
<td>Bam</td>
</tr>
<tr>
<td>abc.com</td>
<td>ama</td>
</tr>
<tr>
<td>abc.com</td>
<td>ma</td>
</tr>
<tr>
<td>abc.com</td>
<td>a h</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>...</td>
<td></td>
</tr>
<tr>
<td>bcd.com</td>
<td>...</td>
</tr>
<tr>
<td>...</td>
<td></td>
</tr>
</tbody>
</table>
Step 2: Compute m Min-hashes

<table>
<thead>
<tr>
<th>docID</th>
<th>mh1</th>
<th>mh2</th>
<th>mh3</th>
<th>mh4</th>
<th>...</th>
<th>mh50 0</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>To</td>
<td>que</td>
<td>lon</td>
<td>or</td>
<td></td>
<td>Not</td>
</tr>
<tr>
<td>2</td>
<td>...</td>
<td>...</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>url</th>
<th>mh1</th>
<th>mh2</th>
<th>mh3</th>
<th>mh4</th>
<th>...</th>
<th>mh50 0</th>
</tr>
</thead>
<tbody>
<tr>
<td>abc.com</td>
<td>sec</td>
<td>ret</td>
<td>def</td>
<td>ens</td>
<td></td>
<td>...</td>
</tr>
<tr>
<td>bcd.com</td>
<td>...</td>
<td>...</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Note: $m = \text{a few hundreds}$
Step 3: Compute Signatures

<table>
<thead>
<tr>
<th>docID</th>
<th>h(mh1,...,mh20)</th>
<th>h(mh21,...,mh40)</th>
<th>...</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2345234</td>
<td>3232</td>
<td>...</td>
</tr>
<tr>
<td>2</td>
<td>...</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>docSIG</th>
<th>docID</th>
<th>Sig</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1</td>
<td>2345234@1</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>3232@2</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>...</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>452342@25</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>23423@1</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>...</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>webSIG</th>
<th>url</th>
<th>Sig</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>abc.com</td>
<td>676876@1</td>
</tr>
<tr>
<td></td>
<td>abc.com</td>
<td>3232@2</td>
</tr>
<tr>
<td></td>
<td>abc.com</td>
<td>...</td>
</tr>
<tr>
<td></td>
<td>abc.com</td>
<td>787892@25</td>
</tr>
<tr>
<td></td>
<td>bcd.com</td>
<td>23423@1</td>
</tr>
<tr>
<td></td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>
Step 4: Find docs with common signatures

```sql
SELECT DISTINCT docSig.docID, webSig.url
FROM docSig, webSig
WHERE docSig.sig = webSig.sig
```

Note: this is a SINGLE JOIN query !!
Still need to filter out the false positives (easy)