Inconsistent databases

• Often arise in data integration.
• Suppose have a functional dependency $name \rightarrow salary$ and two tuples $(John, 10K)$ in source 1, and $(John, 20K)$ in source 2.
• One may want to clean data before doing integration.
• This is not always possible.
• Another solution: keep inconsistent records, and try to address the issue later.
• Issue $=$ query answering.
Inconsistent databases cont’d

• Setting:
  ◦ a database $D$;
  ◦ a set of integrity constraints $IC$ e.g. keys, foreign keys, functional dependencies etc
  ◦ a query $Q$

• $D$ violates $IC$

• What is a proper way of answering $Q$?

• Certain Answers:

$$\text{certain}_{IC}(Q, D) = \bigcap_{D_r \text{ is a repair of } D} Q(D_r)$$
Repairs

• How can we repair an instance to make it satisfy constraints?
• If constraints are functional dependencies: say $A \rightarrow B$ and we have

\[
\begin{array}{ccc}
A & B & C \\
\hline
a1 & b1 & c1 \\
a1 & b2 & c2
\end{array}
\]

we have to delete one of the tuples.

• If constraints are referential constraints, e.g. $R[A] \subseteq S[B]$ and we have

\[
\begin{array}{cc}
A & C \\
\hline
a1 & c1 \\
a2 & c2
\end{array}
\quad
\begin{array}{cc}
B & D \\
\hline
a1 & d1 \\
a3 & d2
\end{array}
\]

then we have to add a tuple to $S$. 
Repairs cont’d

• Thus to repair a database to make it satisfy $IC$ we may need to add or delete tuples.

• Given $D$ and $D'$, how far are they from each other?

• A natural measure: the minimum number of deletions/insertions of tuples it takes to get to $D'$ from $D$.

• In other words,

$$\delta(D, D') = (D - D') \cup (D' - D)$$

• A repair is a database $D'$ so that
  ◦ it satisfies constraints $IC$, and
  ◦ there is no $D''$ satisfying constraints $IC$ with $\delta(D, D'') \subset \delta(D, D')$
How many repairs are there?

Can easily be exponential even for keys: i.e. $\sqrt{2^N}$.

\[
\begin{array}{c|c}
A & B \\
1 & 0 \\
1 & 1 \\
2 & 0 \\
2 & 1 \\
\vdots & \vdots \\
\vdots & \vdots \\
n & 0 \\
n & 1 \\
\end{array}
\]

plus key $A \rightarrow B$

\[
\begin{array}{c|c|c}
A & B & \text{REPAIR} \\
1 & \cdot & \Rightarrow \\
2 & \cdot & \\
\vdots & \vdots & \\
n & \cdot & \\
\end{array}
\]

I.e. for $N = 2n$ tuples we have $2^n = \sqrt{2^N}$ repairs.

(A side remark: this construction gives us $\sqrt{c^n}$ repairs for any number $c$. What is the maximum of $\sqrt{c}$?)
Query answering

- Recall $\text{certain}_{IC}(Q, D) = \bigcap_{D_r \text{ is a repair of } D} Q(D_r)$.
- Computing all repairs is impractical.
- Hence one tries to obtain a rewriting $Q'$:
  
  \[ Q'(D) = \text{certain}_{IC}(Q, D). \]
- Is this always possible?
Query rewriting: a good case

• One relation $R(A, B, C)$
• Functional dependency $A \rightarrow B$
• Query $Q$: just return $R$
• If an instance may violate $A \rightarrow B$, then we can rewrite $Q$ to $R(x, y, z) \land \forall u \forall v (R(x, u, v) \rightarrow u = y)$ or

$$\text{SELECT } * \text{ FROM } R \text{ WHERE NOT EXISTS (SELECT } * \text{ FROM } R \text{ R1 WHERE R.A=R1.A AND R.B <> R1.B)}$$

• This technique applies to a small class of queries: conjunctive queries without projections, i.e.

$$\text{SELECT } * \text{ FROM R1, R2 ... WHERE <conjunctio} \text{n of equalities>$$
Query rewriting: a mildly bad case

• One relation $R(A, B)$; attribute $A$ is a key
• Query $Q = \forall x, y, z ((R(x, z) \land R(y, z)) \rightarrow x = y)$
• Question: is $Q$ true in all repairs?
• Another way of looking at it: if $Q'$ is the negation of $Q$:
  o $Q' = \exists x, y, z (R(x, z) \land R(y, z) \land (x \neq y))$
  o Question: is $Q'$ false in all repairs?
• This happens precisely when $R$ contains a perfect matching
• But checking for a perfect matching cannot be expressed in SQL.
• Hence, no SQL rewriting for certain $IC(Q)$. 
Query rewriting: the worst

- One can find an example of a rather simple relational algebra query $Q$ and a set of constraints $IC$ so that the problem of finding

$$\text{certain}_{IC}(Q, D)$$

is coNP-complete.

- In general for most types of constraints one can limit the number of repairs but they give rather high complexity bounds
  - typically classes “above” PTIME and contained in PSPACE – hence almost certainly requiring exponential time.
Other approaches

• Repair attribute values.
  ◦ A common example: census data. Don’t get rid of tuples but change the values.
  ◦ Distance: sum of absolute values of squares of differences new value – old value
  ◦ Typically one considers aggregate queries and looks for approximations or ranges of their values

• A different notion of repair.
  ◦ Most commonly: the cardinality of \((D - D') \cup (D' - D)\) must be minimum.
  ◦ This is a reasonable measure but the complexity of query answering is high.