Co-Evolution of Data and Applications

by

Michael Stonebraker, Dong Deng and Michael Brodie

DBg Database Group
MIT Computer Science and Artificial Intelligence Lab

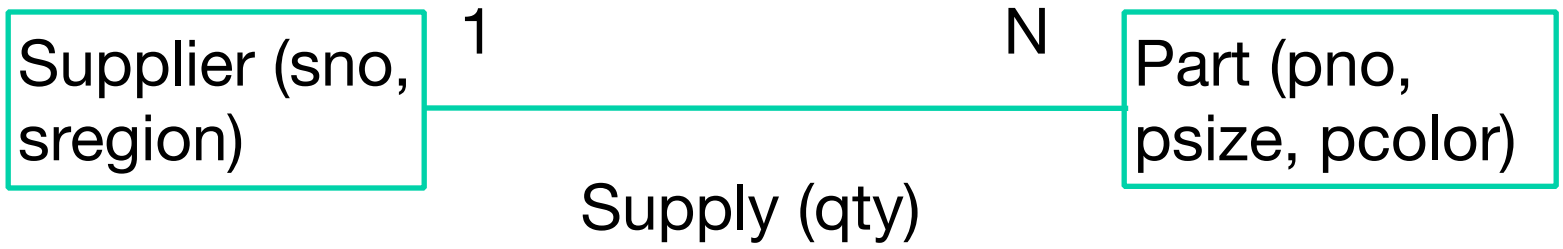
Our Thesis

- Traditional database design is all wrong
  - Gold standard techniques are not used “in the wild” and for good reasons
  - Contributes to database decay
- If you used the gold standard techniques…
  - You would get application decay
- Probably better to take your poison in both dimensions

Traditional Database Design Wisdom

- Use a modeling tool (e.g. an Entity-Relationship diagram tool)
- To construct/modify an E-R diagram
- When satisfied, push a button
- Tool spits out a relational schema in 3rd normal form
- Code in ODBC/JDBC against this schema

Example



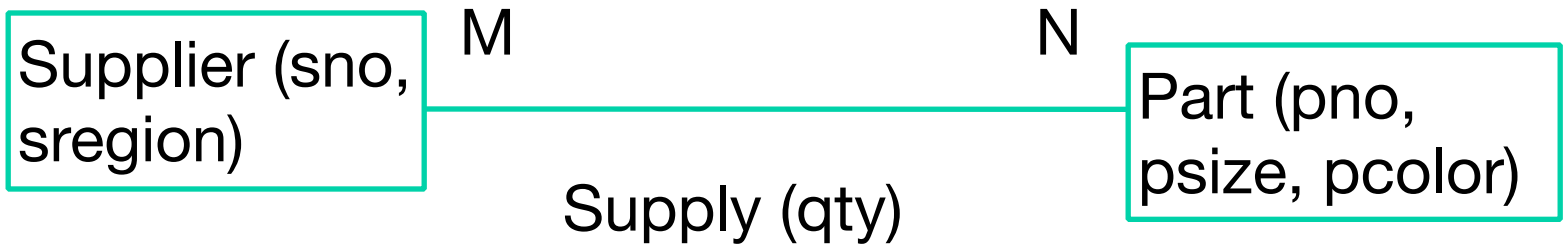
Two entities (Supplier and Part with attributes)
One relationship (Supply with attributes, which is 1 – N)

Code in ODBC/JDBC for the resulting tables

Now Suppose….

- Management decides to allow multiple suppliers for each part, as long as they are in different regions
  - Perhaps to get better terms
  - Or …
- Such changes happen frequently “in the wild”
  - Often once a quarter or more

New Diagram



Two entities (Supplier and Part with attributes)
One relationship (Supply with attributes which is now M – N)

New tables to code against

Summary of The Traditional Wisdom

- Convert “old” database to “new”
- Define “old” as views
- Applications (in theory) continue to run
- Database stays in 3NF (defined as goodness by research community)

A Dirty Little Secret

- Based on a survey of about 20 DBAs at 3 large companies…..
  - None use this methodology

    or
  - Use it for "green field" design and then abandon it
- I.e. the "gold standard" is not used "in the wild"

Why Not????

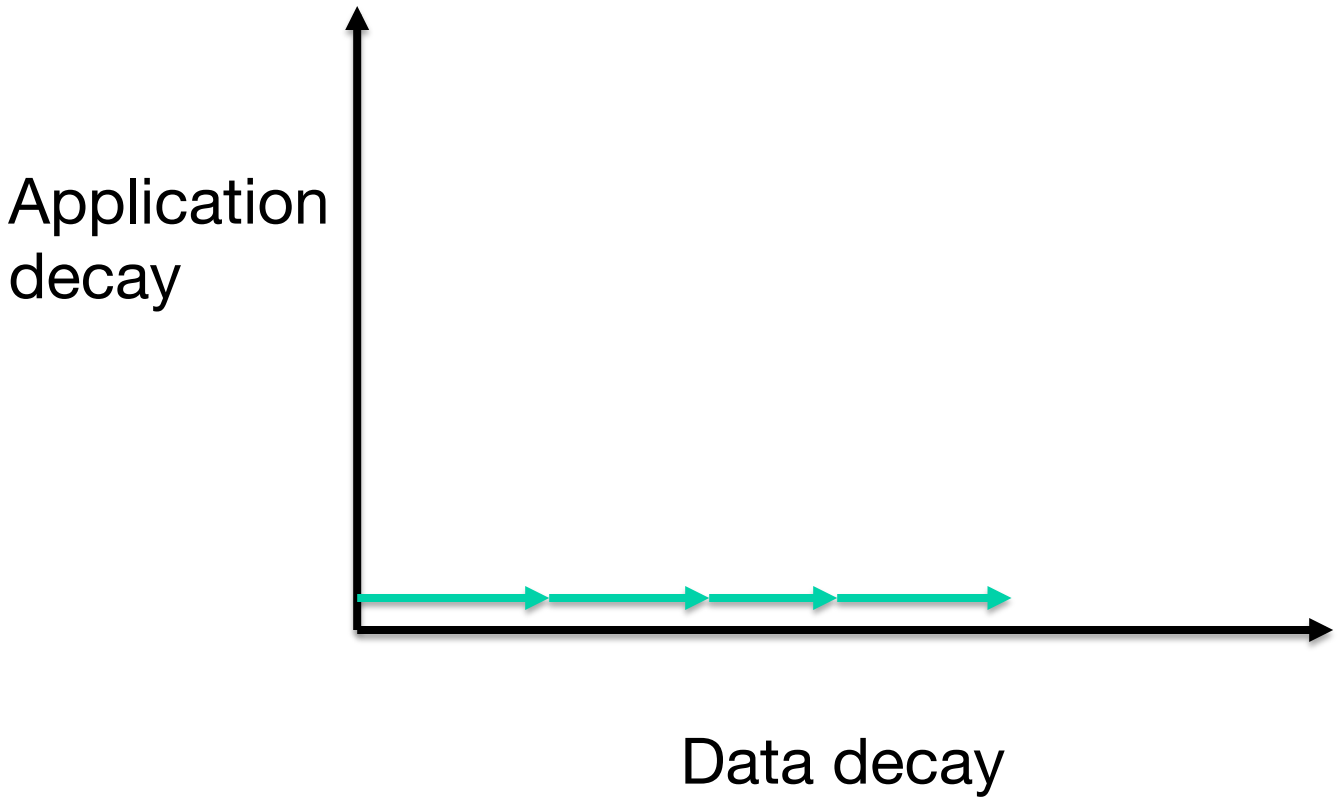
- View update problems
- Semantics change – often requires recoding
  - As an aside: few DBAs make any “optional” changes
  - And most research deals with “non semantic” changes

- Net result -- Substantial risk!
  - Applications all over the enterprise must be found and corrected
  - Often with no budget for this activity

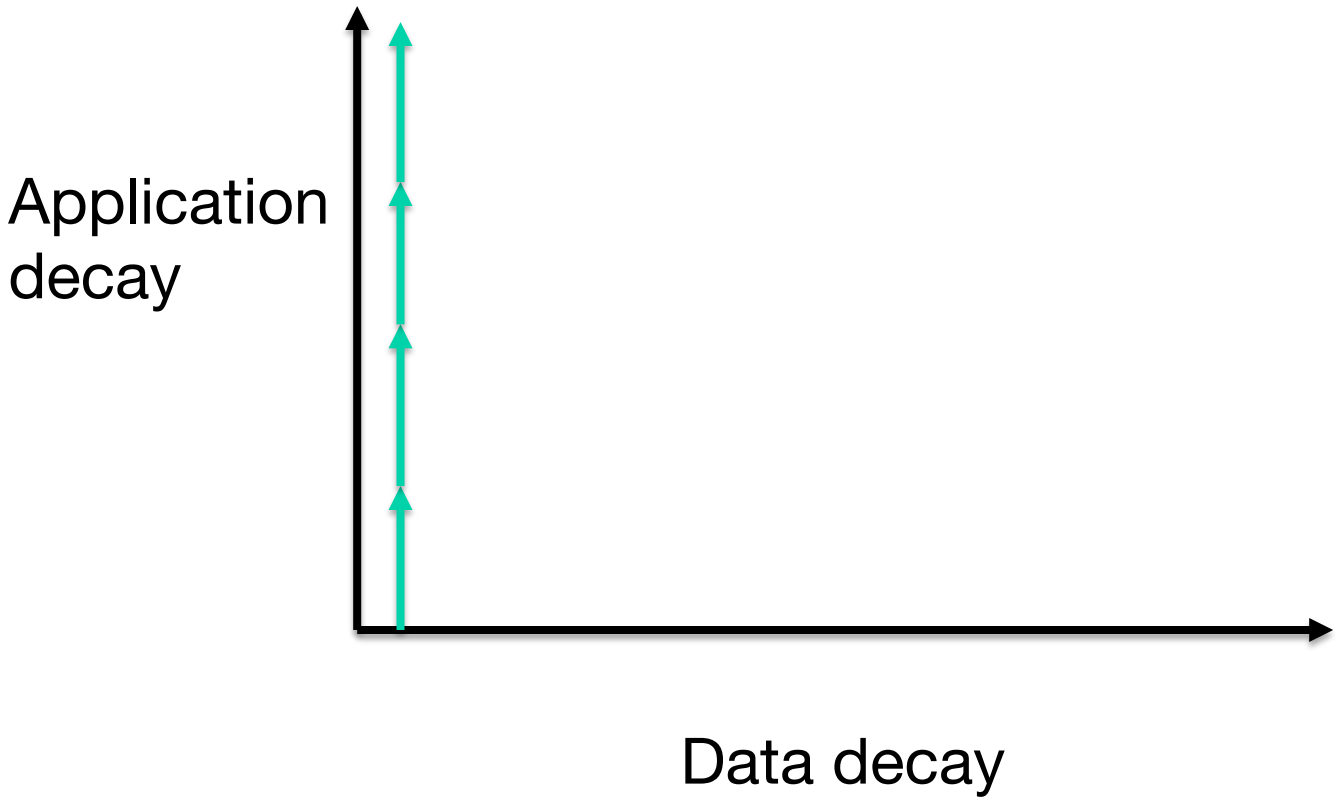
Less Risky Solution -- Kluge

- Leave the old schema as is
  - And kluge the semantics
  - Often lowers or removes application maintenance!!!!
- But the result does not conform to any ER diagram!
  - No longer 3NF
- Database decays over time!!!!!
  - i.e drifts further and further from 3NF

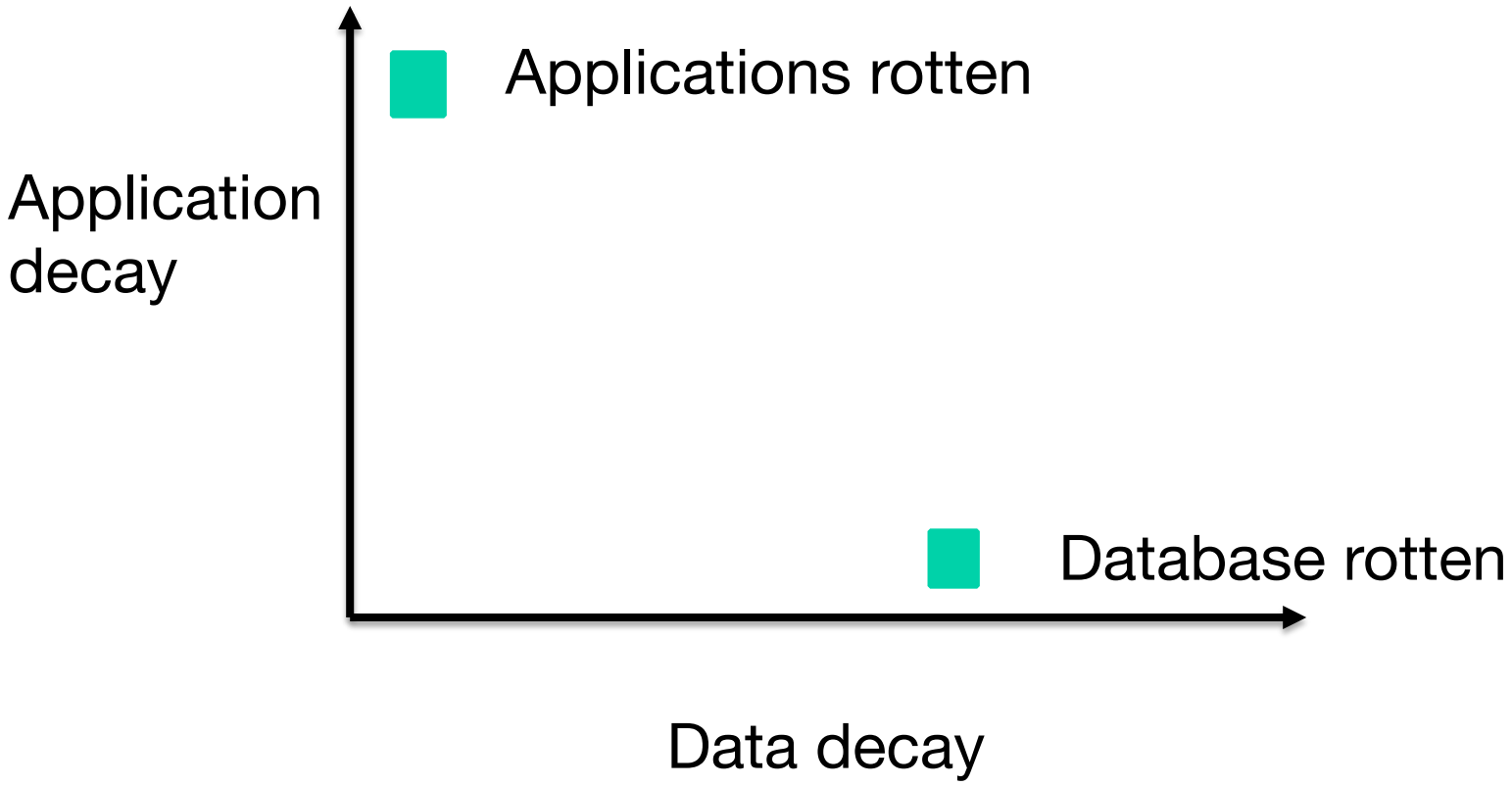
Looked at Graphically Over 4 Evolutions



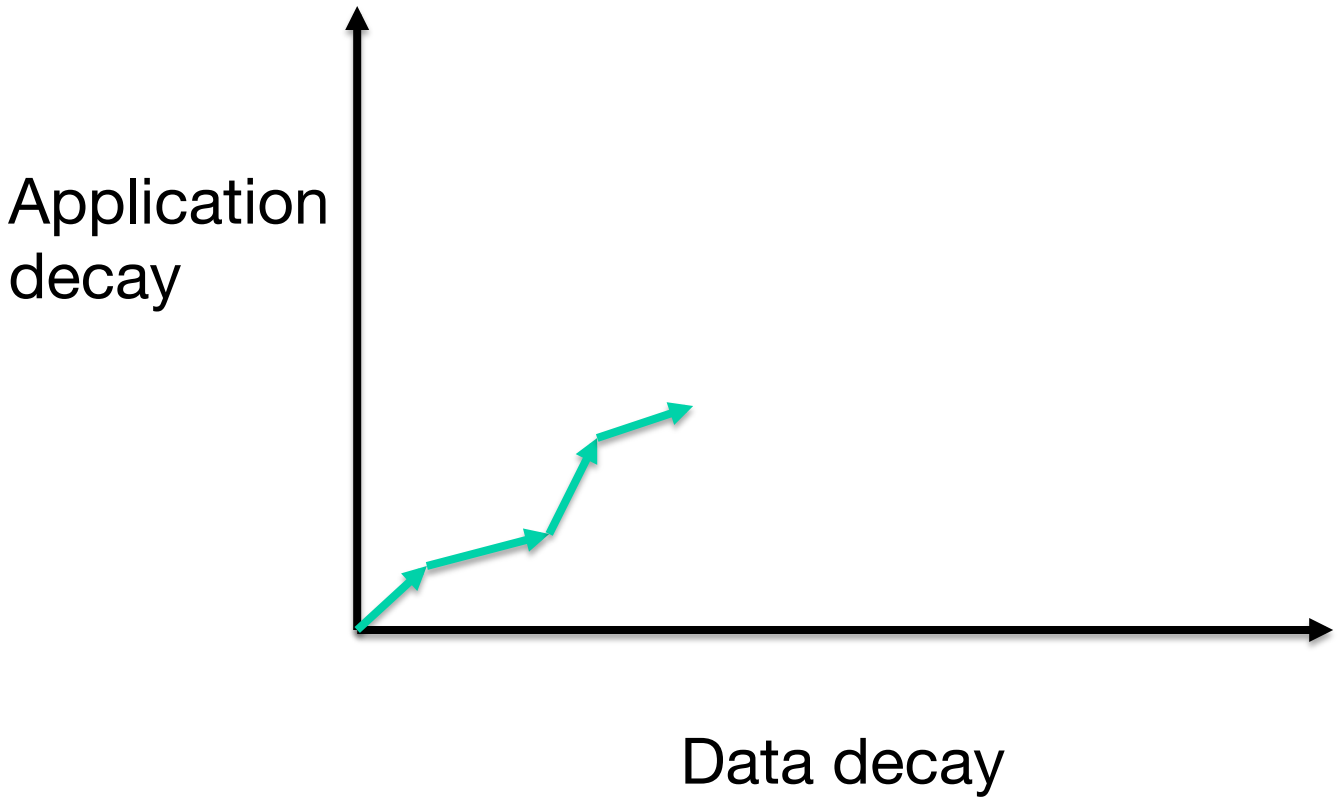
Suppose You Followed the DBMS Gold Standard….



Choices So Far…..



Our Thesis – Co-Evolution Likely to Result in a Better Solution



We Have….

- An architecture to support co-evolution
  - Cannot allow application groups to independently code in ODBC/JDBC
- Composite cost function for overall decay
  - I.e. evolution cost
- Algorithms to monitor database decay
- Working on algorithms to monitor application decay
- Working on algorithms to suggest schema chages

HELP!!!!!!!

- We need a real-world DBMS application to try our stuff on
- We need:
  - Your schema and application code
  - Over a few changes