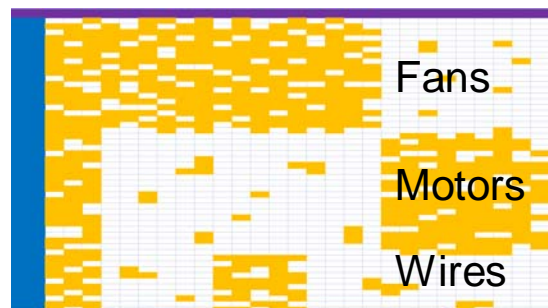
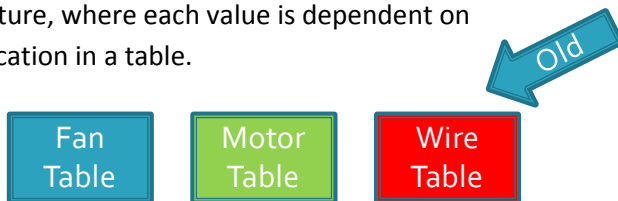


# Legume Lowering the Barriers to Data Management

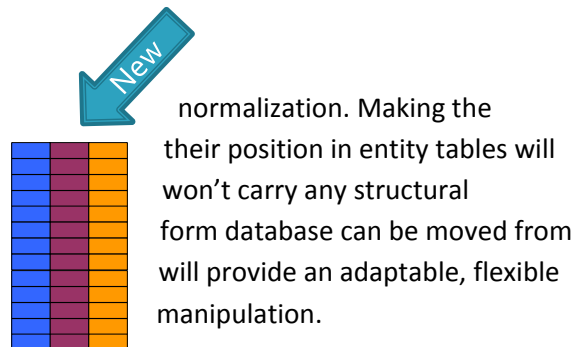
- ◆ Revitalize your relational database with new organization data, even objects, using a handful of tables, and relate them to each other with **standard SQL** commands
- ◆ Store a **mixture of databases** and models together in a **uniform table design**. These tables act together as a flexible relational container that **doesn't rely on XML** or user-defined blobs.
- ◆ **Maintain** the **SQL database training** and tools that are standard to your organization and **improve learning curves** using standard methods.
- ◆ Use a **scalable**, data-neutral structure to **remove the gridlock** from architecture, development interaction and conflicting business needs.
- ◆ **Act quickly** to handle new business opportunities with an agile-friendly design.
- ◆ **Improve reliability** using tested reusable components that can be configured to handle custom needs.
- ◆ Store **legacy data**, **tracking data**, and data **warehouses** in a standardize container available for **data mining** and relationship building.
- ◆ Use uniform storage containers to lower the cost of **point-to-point** transactions, data integration, and data distribution.
- ◆ **Merge data** from different databases while preserving the **original organization** models.
- ◆ Manage new data into your database **without structural changes**.

## What are Equal Format Databases?

Relational databases are traditionally designed using a normalization method that generates different tables for each entity and relationship type. This led every database to be custom-designed to a model-specific structure, where each value is dependent on its location in a table.



Equal Format Databases takes a different approach to assumption that the values should be independent of lead to a design that becomes model-neutral and dependence from the data that it carries. This equal container to container in a more fluid manner and container that will support reusable software



normalization. Making their position in entity tables will won't carry any structural form database can be moved from will provide an adaptable, flexible manipulation.

## Feature Comparison

Equal Format Design can be applied to any storage system but it is typically implemented in relational database systems. This chart compares a typical relational implementation of Equal Format Design to typical implementations of other database methodologies and systems.

Feature	Equal Format Design <sup>1</sup>	Traditional Relational Design <sup>2</sup>	Object Database	XML Storage	XML Text File
Uses any standard Relational DBMS server, such as IBM DB2, Oracle, MS SQL Server, MySQL	Yes	Yes	Rarely	Rarely	No
Cross-platform storage	Yes	Yes	Rarely	Rarely	Yes
SQL Data Access	Yes	Yes	Rarely	Rarely	No
Values indexed	All	Some	Some	Some	None
“Columns” indexed <sup>3</sup>	Yes	Yes	Some	Some	No
Dynamic “column” selection <sup>4</sup>	Yes	No	No	Rarely	Rarely
Multi-user editing	Yes	Yes	Yes	Yes	No
Track or compare all changes by user, project, time period, etc.	Yes	Difficult <sup>5</sup>	Difficult <sup>5</sup>	Difficult <sup>6</sup>	Difficult <sup>6</sup>
Non-destructive changes	Yes	Difficult <sup>5</sup>	Difficult <sup>5</sup>	Difficult <sup>6</sup>	Difficult <sup>6</sup>
Point-in-time selection	Yes	Difficult <sup>5</sup>	Difficult <sup>5</sup>	Difficult <sup>6</sup>	Difficult <sup>6</sup>
Global keyed value data system	Yes	Difficult <sup>5</sup>	Difficult <sup>5</sup>	Inefficient <sup>7</sup>	Inefficient <sup>7</sup>
Reusable structure and transport	Yes	No	No	Yes	Yes
Reusable software and methods	Yes	No	No	Yes	Yes
New “columns” accommodated without structural changes <sup>3</sup>	Yes	No	No	Yes	Yes
Relational Data Model	Yes	Yes	Limited <sup>8</sup>	Inefficient <sup>10</sup>	Inefficient <sup>11</sup>
Object Data Model	Yes	Limited <sup>9</sup>	Yes	Inefficient <sup>10</sup>	Inefficient <sup>11</sup>
Hierarchical Data Model	Yes	Yes	Yes	Inefficient <sup>10</sup>	Inefficient <sup>11</sup>
Mixed Hierarchical Data Model	Yes	Yes	Sometimes	Sometimes	Sometimes
Knowledgebase Data Model	Yes	Limited <sup>12</sup>	Limited <sup>12</sup>	Limited <sup>12</sup>	Limited <sup>12</sup>
Semantic Data Model	Yes	Limited <sup>12</sup>	Limited <sup>12</sup>	Limited <sup>12</sup>	Limited <sup>12</sup>
Semantic Web Data	Yes	Limited <sup>12</sup>	Limited <sup>12</sup>	Limited <sup>12</sup>	Limited <sup>12</sup>
Mixed Data Models	Yes	Limited <sup>12</sup>	Limited <sup>12</sup>	Limited <sup>12</sup>	Limited <sup>12</sup>

<sup>1</sup> A relational database system using Equal Format Design tables

<sup>2</sup> A traditional relational database design using E-R modeling and normalization design techniques

<sup>3</sup> Columns are treated the same as values in some systems

<sup>4</sup> Dynamic column selection means “columns” can be selected with the same syntax used to select a row in a table

<sup>5</sup> Row storage format makes it difficult or inefficient to manage individual cells

<sup>6</sup> Object field mapping makes it difficult to handle multiple copies of fields

<sup>7</sup> The global key system in XML has long namespaces

<sup>8</sup> Object models encapsulate fields, making it difficult to search fields

<sup>9</sup> Traditional relational techniques make it difficult to relate any object to any object

<sup>10</sup> Database holds most data in streams, limiting the efficiency of data access

<sup>11</sup> Text holds the data in a stream, limiting access methods

<sup>12</sup> Normally limited to triplet expressions or techniques

<sup>13</sup> Limits caused by model incompatibilities with storage structure

