

Accessing Your Enterprise Geodatabase through SQL

Amit Kesarwani and Travis Val

March 12, 2015

Assumptions

- **Basic knowledge of SQL and relational databases**
- **Basic knowledge of the geodatabase**
- **We'll hold all questions until the end**

Please turn off cell phones



Agenda

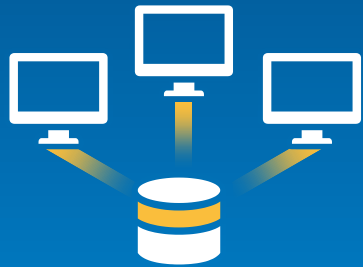
- **RDBMS support in ArcGIS**
- **Geodatabase and it's system tables**
- **Demo - Querying geodatabase schema**
- **Editing geodatabase feature classes from SQL**
- **Discovery functions**
- **Demo - Editing versioned and non versioned feature class**
- **Summary of dos and don'ts**

ArcGIS Supports Multiple Implementation Patterns

- Leveraging Common Computing Architecture



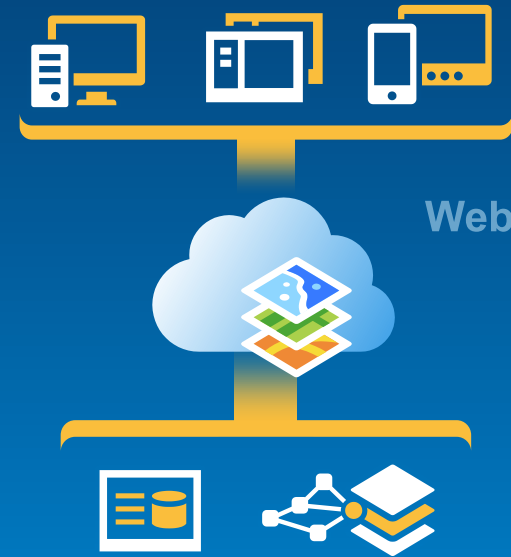
File Based



Database Centric



Server Centric



Web Centric

Web GIS

Database

- **You can access spatial or non-spatial data in a DBMS to use in ArcGIS**
 - **Geodatabase and Database access**
 - Oracle, SQL Server, DB2, Informix, PostgreSQL
 - **Database access only**
 - Netezza, Teradata, SAP HANA, ALTIBASE, Windows Azure SQL Database
- **You can connect directly to a supported database and view data by making a connection from the Catalog tree in ArcGIS for Desktop**
- **To filter what data appears in ArcMap, you can use a query layer**
- **Use SQL to access the data**

Accessing spatial data from ArcGIS

- A table with a column that stores a spatial type
 - We call this a feature class
- Each row represents a feature
- The fields in each row represent attributes or properties of the feature
- One of the fields holds the feature geometry which is stored as a spatial type

Parcels							
	OBJECTID *	SHAPE *	PROPERTY_I *	Res	Zoning_simple	SHAPE_Length	SHAPE_Area
	1	Polygon	5001	Non-Residential	<Null>	3597.780813	112552.418591
	2	Polygon	5002	Non-Residential	<Null>	814.855837	18488.417709
	3	Polygon	1003	Residential	Residential	489.655523	12815.591379
	4	Polygon	1004	Residential	Residential	521.761248	14036.135346
	5	Polygon	1005	Residential	Residential	453.479649	9816.352665



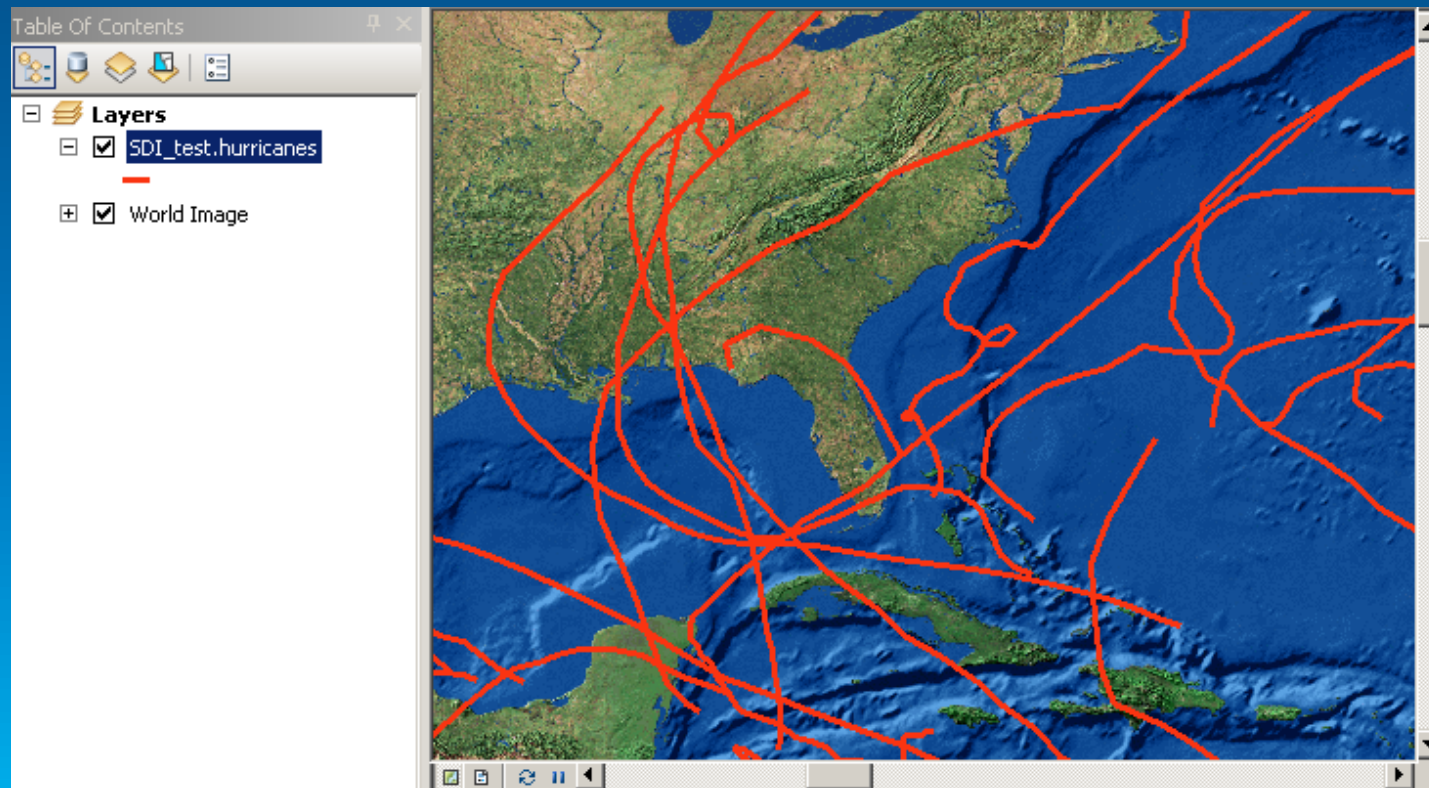
Viewing database data in ArcGIS

- **Tables (with and without a spatial type) are viewed in ArcGIS through a query layer**
 - Define the layer yourself or let ArcGIS discover how to define it
- **Query Layer is a layer that is defined by a SQL query**
 - Provide data integration with geodatabases as well as from databases
 - Can quickly integrate spatial and non-spatial information into GIS projects independently of where and how that information is stored

Viewing database data in ArcGIS

- Simple SQL query

```
SELECT ObjectID, Name, Shape FROM dbo.HurricaneTracks_2005 hurricane
```



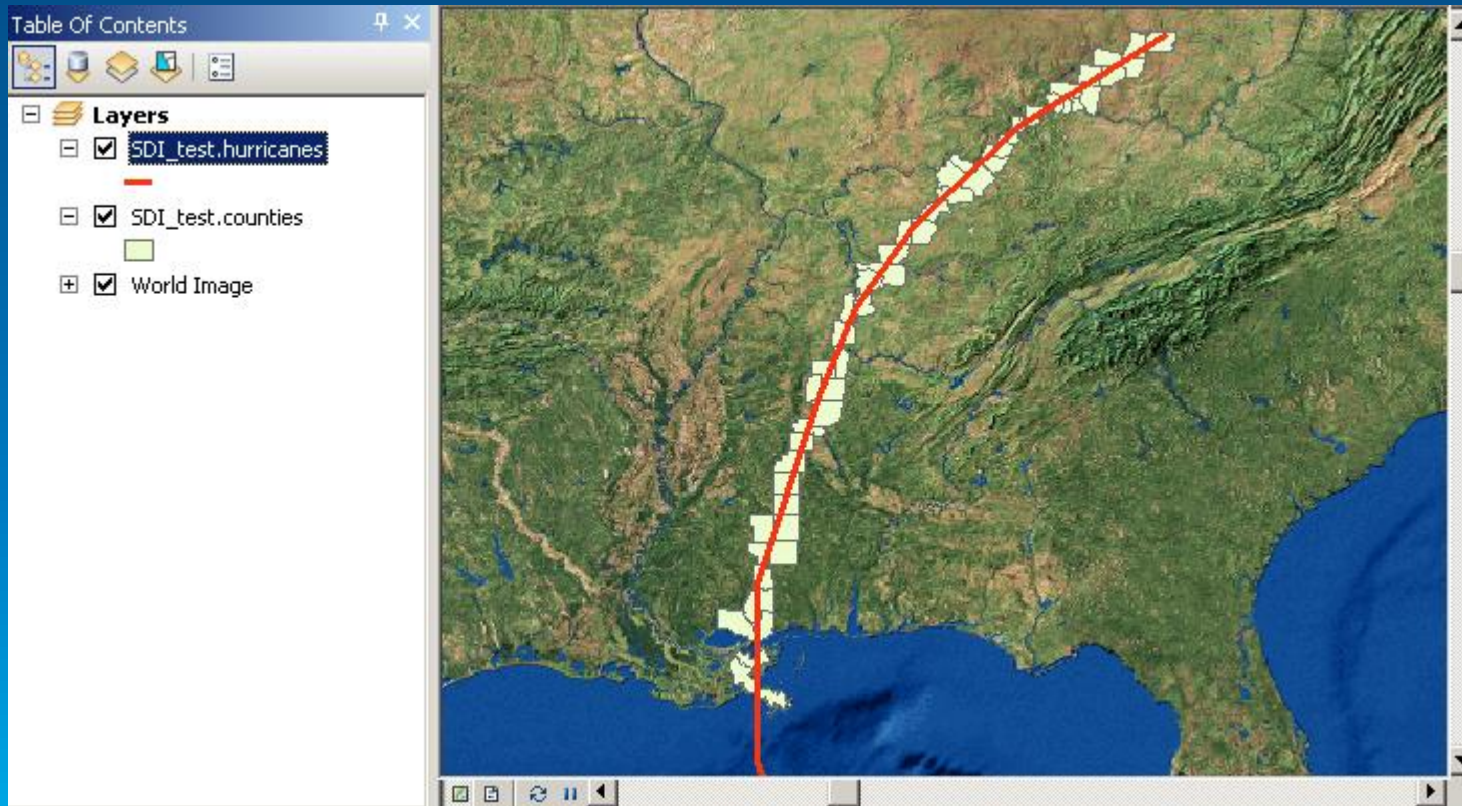
Viewing database data in ArcGIS

- **Complex queries that use casting, derived columns and spatial operators**

```
SELECT county.id, county.State_name,  
county.NAME county_name, county.POP1990 population,  
CAST(county.POP1990 as decimal)/CAST(states.POP1990 as decimal)*100 PctStatePop,  
county.Shape FROM dbo.HurricaneTracks_2005 hurricane  
join dbo.counties county  
on hurricane.Shape.STIntersects(county.shape) = 1  
Join dbo.states states  
on states.state_fips = county.state_fips  
WHERE hurricane.NAME = 'KATRINA'
```

Viewing database data in ArcGIS

- Complex queries that use casting, derived columns and spatial operators



Other Database Tasks

- **Connecting to a database**
- **Supported data types**
- **Viewing data and query layers**
- **Some administration tasks**
 - **Managing permissions**
- **Create new tables and alter schema**

Building on top of database functionality

- **When you want to do more with your data**
 - **Store business rules with the data so they're available to everyone who accesses the data**
 - **Advanced data modeling like transportation or utility networks**
 - **Store and work with detailed cartography**
 - **Multiple editors working on the same data at the same time without impacting each other.**

The Geodatabase

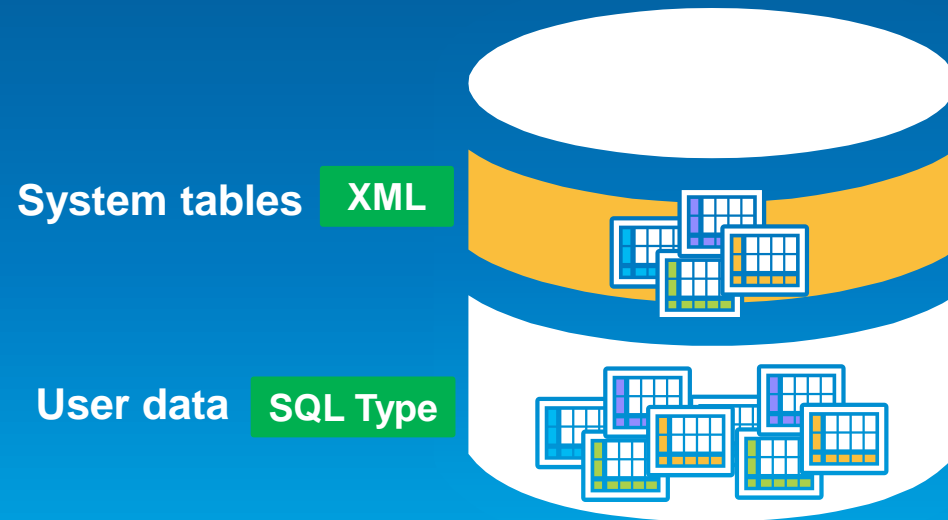
- **A physical store of geographic data**
 - Scalable storage model supported on different platforms
- **Core ArcGIS information model**
 - A comprehensive model for representing and managing GIS data
 - Implemented as a series of simple tables
- **A transactional model for managing GIS workflows**
- **APIs for accessing data**

Geodatabase is based on relational principles

- **The geodatabase is built on a relational database**
 - **Functionality consistent across each enterprise DBMS**
- **Database provides:**
 - **Storage of geographic data in tables**
 - **Extended functionality and data integrity**
- **Application logic provides:**
 - **GIS integrity and behavior**
 - **Business rules, topology, networks, etc.**

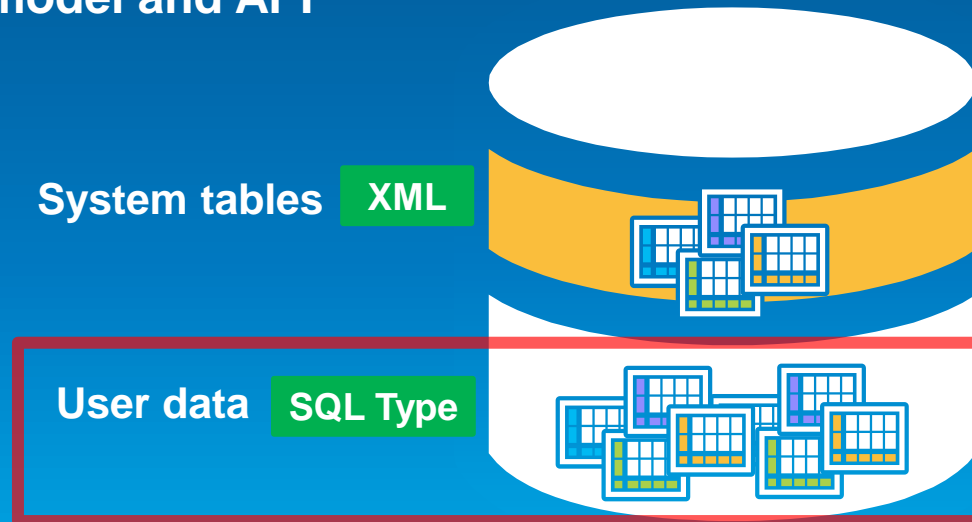
Geodatabase schema

- There are two sets of tables
 - Dataset tables (user-defined)
 - Geodatabase system tables (schema is controlled by ArcGIS)



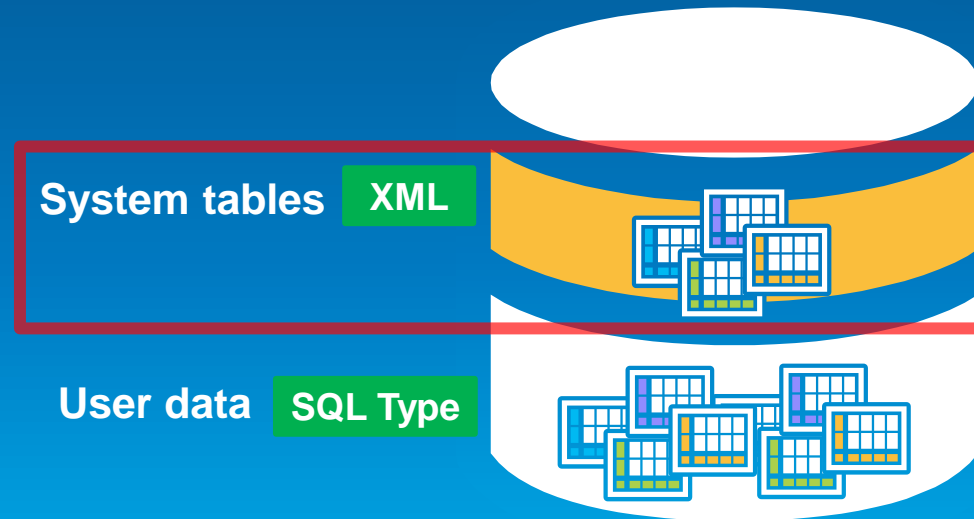
User-defined tables

- Stores the content of each dataset in the geodatabase
- Datasets are stored in 1 or more tables
- Spatial types enhance the capabilities of the geodatabase
 - SQL access to geometry
 - Industry standard storage model and API

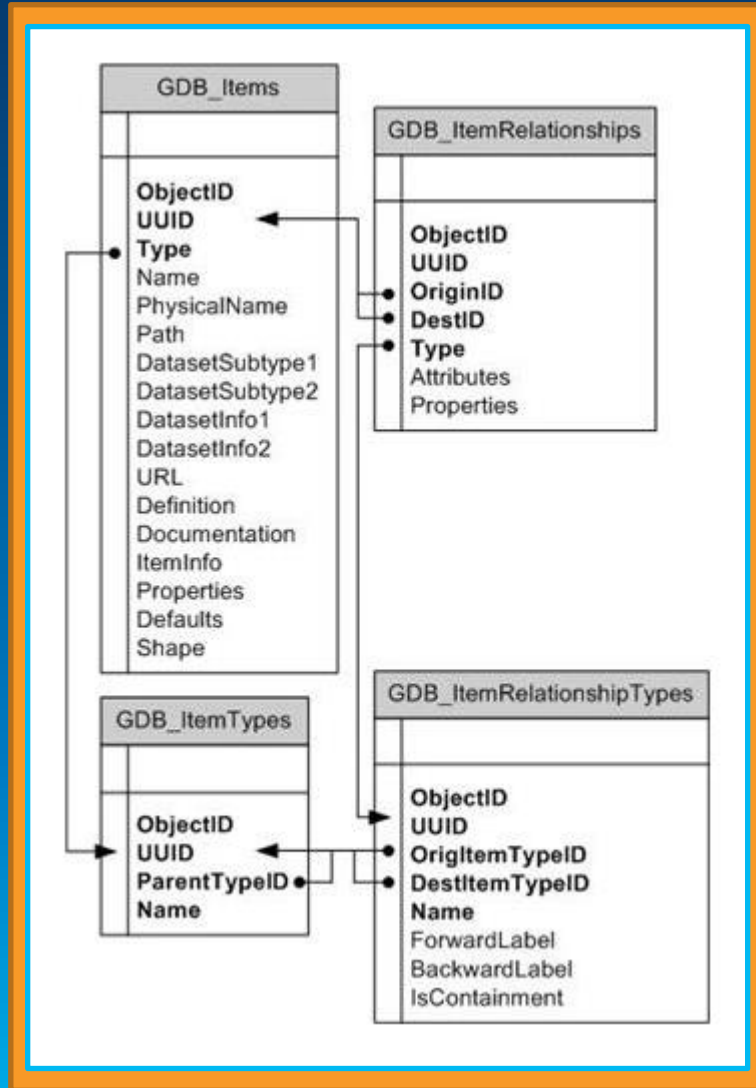


Geodatabase system tables

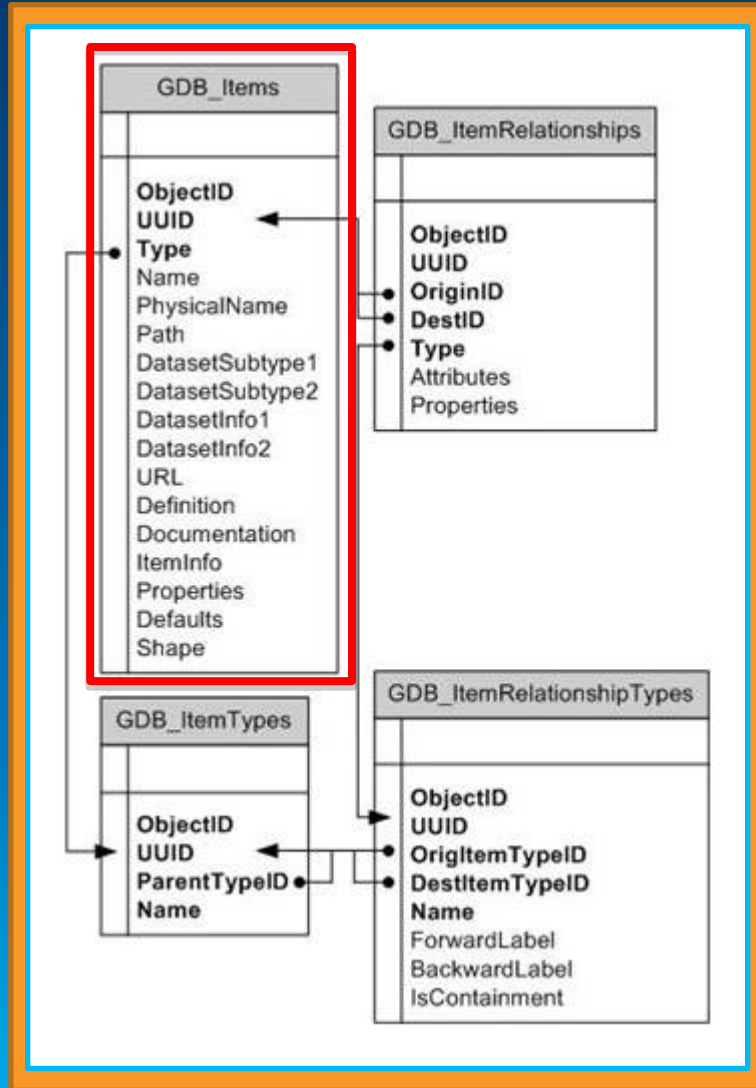
- System tables store definitions, rules and behavior for datasets
- Tracks contents within a geodatabase
- 4 main system tables
- Geodatabase schema is stored primarily within an XML field



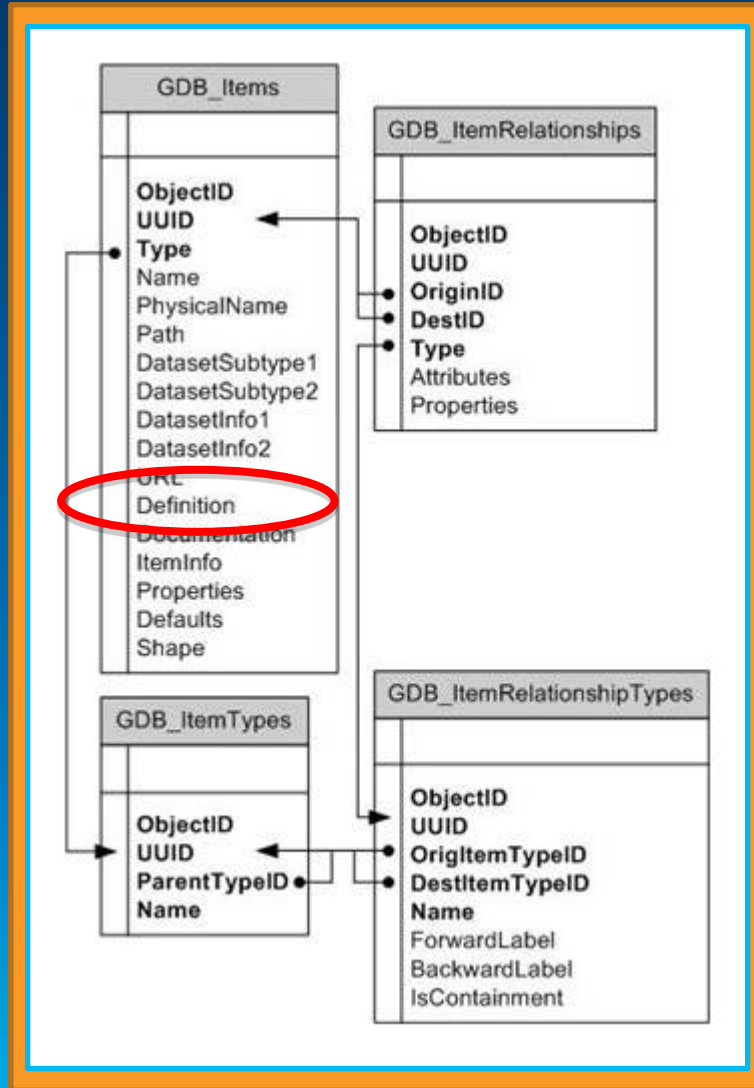
Geodatabase schema



Geodatabase schema

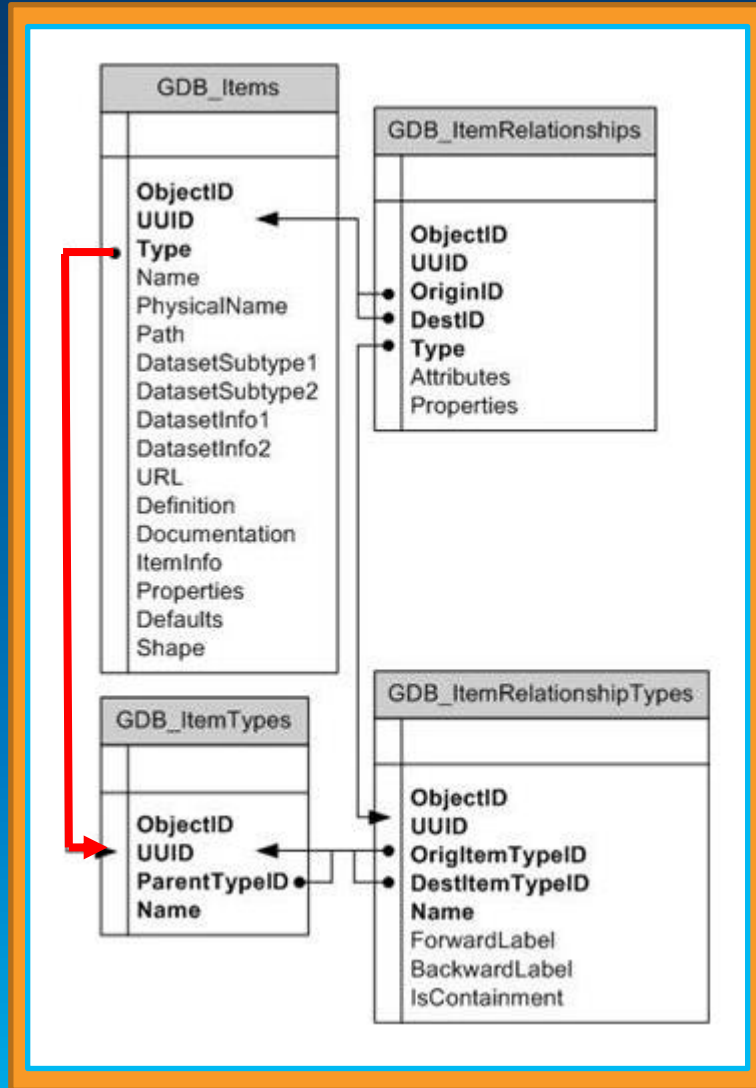


Geodatabase schema

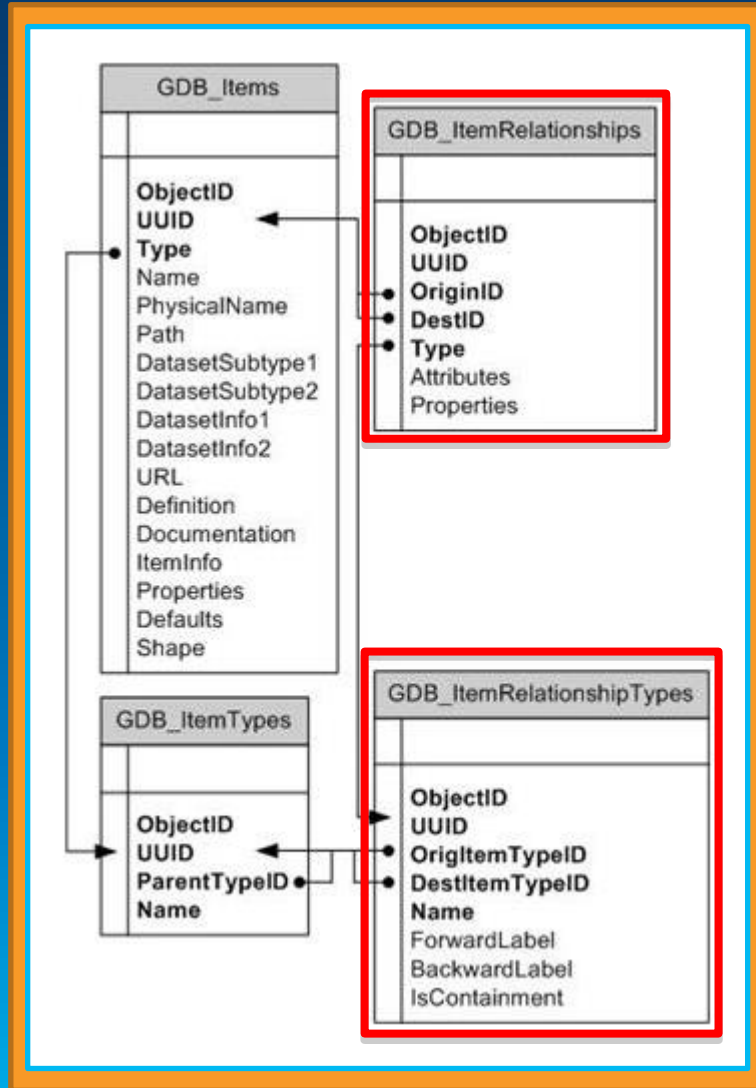


- SQLServer, PostgreSQL and DB2 are native xml types.
- Oracle and Informix use ArcSDE xml

Geodatabase schema



Geodatabase schema

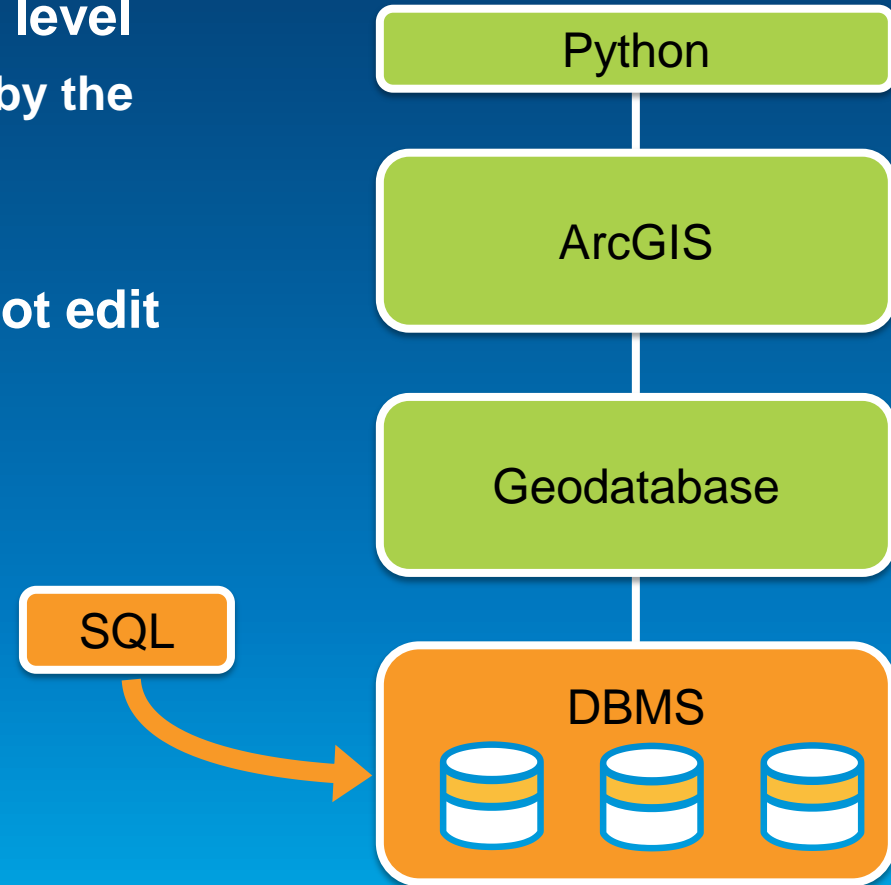


Accessing your geodatabase using SQL

- Access schema and properties of existing datasets
 - Use SQL statements to query the *definition* field on the *gdb_items* table
 - Oracle and Informix use *Gdb_Items_vw*
- Editing tables/feature classes, whether versioned or not
 - Use versioned views with versioned classes
- Create tables with SQL that contain spatial or raster types
- Leverage SQL functions to evaluate attributes and spatial relationships, perform spatial operations, and return and set spatial properties.

Accessing your geodatabase using SQL

- **With SQL, you access the data at the DBMS level**
 - Bypass behaviors and functionality enforced by the geodatabase or ArcGIS clients
- **Need to be aware of what you can and cannot edit**
 - Relationship classes
 - Geometric networks
 - Topology...



Querying the geodatabase schema

- Look at system tables
- Relationships
- Domain references



What is a spatial type?

- **A type that stores geometry data in a single spatial attribute**
 - Geometry type, coordinates, dimension, spatial reference
- **Spatial index**
 - Improves spatial searches
- **Relational and geometry operators and functions**
 - Constructors – creates new geometry
 - Accessor – return property of a geometry
 - Relational – perform spatial operations
 - Geometry – transform from one geometry to another

Benefits of a spatial type

- **With SQL and a spatial type you can**
 - Create tables with a spatial attribute
 - Read and analyze spatial data
 - Insert, update and delete **simple** features
- **Enhances efficiency**
 - Data and methods are stored in the database
 - Applications access native dbms type
- **Access using common APIs and SQL**
 - Standard functions
 - Well-known interchange formats

Creating geodatabase feature classes using SQL

- Use SQL to create and populate tables

```
CREATE TABLE hazardous_sites  
  (oid INTEGER NOT NULL, site_id INTEGER,  
   name VARCHAR(40), location sde.st_geometry)
```

- Need to register the table with the geodatabase to participate in geodatabase functionality
- Once registered, you cannot modify table schema (e.g. add a field) through SQL
 - DML is okay, DDL is not okay

Editing geodatabase feature classes using SQL

- What can you edit?
 - Simple features (points, lines, polygons)
 - Without geodatabase behavior
 - Use the *Is_Simple* function to determine whether your data can be edited
- Editing non-versioned tables
 - Edit tables directly
- Editing versioned tables
 - Edit special *versioned view* instead of tables

Editing the ObjectID field

- Every geodatabase feature class or table has an ObjectID field
 - Unique, not null integer
- Value is generated and maintained by ArcGIS
- Non-versioned inserts
 - Use *RowID_Name* to determine name of ObjectID field
 - Use *Next_RowID* function to obtain next ObjectID value
- Versioned inserts
 - ObjectID obtained automatically by versioned view
- Never *update* an ObjectID field

Editing versioned tables and feature classes

- **Versioning**
 - Supports concurrent editing with long transactions
 - Undo/redo experience
 - No locking or data extraction required
- All changes written to delta tables
 - Adds (a) table and Deletes (d) table
- Edits are assigned an identifier (state_id)
- A version references a lineage of state_ids

Versioned views

- **Specialized view to work with versioned data using SQL**
 - **Uses stored procedures, triggers and functions to access and edit versioned tables**
- **Result set is based on versioned query**
- **Created on a single versioned table, contains all columns**
- **Created automatically when feature class or table is versioned**
- **Versioned must be reconciled through ArcGIS**

Discovery Functions

- **Is_simple** – returns false/true or not registered with Geodatabase
- **Is_Versioned** - returns false/true or not registered with Geodatabase
- **Version_View_Name** - returns the name of the versioned view defined on that table
- **Is_Archive_Enabled** - determines if the specified table has been enabled for archiving.
- **Archive_View_Name** – returns name of the archive view on a nonversioned, archive-enabled table
- **Is_Replicated** - returns false/true
- **Geometry_Columns** - returns the name of all spatial columns in the table that is not registered with Geodatabase
- **GlobalID_Name** - returns the name of the global ID field in the table.
- **RowID_Name** returns the name of a table's ObjectID (or RowID) field

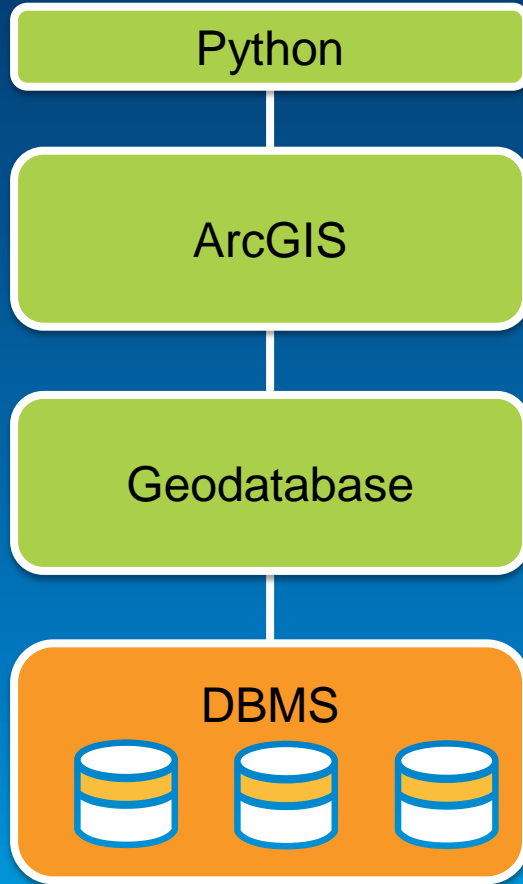
Demo

Editing

Discovery functions
Non-versioned edits
Versioned view edits



Guidelines for using SQL and the geodatabase



- Understand the geodatabase system tables and their structure
- Avoid changing data that affects geodatabase software level behavior
- Geodatabase awareness
 - You have it
 - The database does not

Guidelines for using SQL and the geodatabase

- Do perform spatial operations
- Do query spatial and attribute information
- Do INSERT, UPDATE and DELETE geometries
 - As long as you pay attention to behavior
- Do INSERT, UPDATE and DELETE attribute data
 - As long as you pay attention to behavior
- Do write efficient SQL

Guidelines for using SQL and the geodatabase

- DO NOT update the objectid (row_id) value
- DO NOT modify geometries for feature classes participating in
 - Topologies, geometric networks, network datasets, terrains, parcel fabrics, geodatabase replication, schematic datasets, feature-linked annotation...
- DO NOT update attributes that define geodatabase behavior
 - Enable/Disabled attributes, ancillary attributes, weight attributes...
- Use *Is_Simple* to check

Resources

- **Comprehensive documentation covering**
 - Accessing dataset properties
 - Editing geodatabase data
 - Esri spatial and raster type reference
- **Get started at:**
 - <http://desktop.arcgis.com/en/desktop/latest/manage-data/using-sql-with-gdbs/sql-and-enterprise-geodatabases.htm>

Online survey

- Please give us your feedback

<http://www.esri.com/RateMyDevSummitSession>



Understanding our world.