Proven Database Migration Strategies
Agenda

Speaker Introductions
Overview of Data Lifecycle Management
Database Migration Process
   Issues with Cross-Platform Migration
   Tips and Pitfalls to Avoid
Speaker Introductions

Robin Schumacher
VP Product Management – Embarcadero Technologies
17 years experience as DBA/developer/ETL
Former software reviewer/writer for DM Review, eWeek, Intelligent Enterprise, and others
Author of three database performance books

Billy Bosworth
Director of Development Tools – Embarcadero Technologies
10+ years experience as database developer/DBA
Author of database development book
Enterprise Data Lifecycle Management

Enablers:
- Data Architects
- DBAs
- Developers
- ETL Specialists
- Performance Analysts

Access:
- Data Management Service Layer
- Database Management Service Layer
- Performance Management Service Layer

Provisioning:
- Application Systems
- Business Reporting
- Ad-hoc Requests

Data Design
- RDBMS
- Files
- XML
Database Migration Process

Data Migration Lifecycle

1. Document Sources
   - User Accounts
   - Entities
   - Relationships
   - Datatypes
   - Rules (data, etc.)

2. Design Targets
   - User accounts
   - Entities
   - Relationships
   - Datatypes
   - Rules (data, etc.)

3. Design ETL Flows
   - Simple moves
   - Data Transformations
   - Aggregations
   - Data Profiling/Cleansing

4. Build/Run Migrations
   - ETL Runs
   - Scheduled/Recurring
   - Validation
   - Migrate Code?

5. Performance Test
   - Simulation
   - Metric Analysis
   - Review

- Scripts/Manual
- Data Models
- Models, Reports, etc.

- Scripts/Manual
- Data Models
- Models, Reports, etc.

- Scripts/Vendor Supplied
- Tactical 3rd Party
- Enterprise Global

- Scripts/Manual
- Tactical 3rd Party
- Enterprise Global

- Manual
- Tactical 3rd Party
- Automated
Step 1 – Document Sources

Need

- Owners/User Accounts
- Entities
- Relationships
- Datatypes
- Rules (Ref Integrity, domains, etc.)

Manually reverse engineer, but hard when no data dictionary …

Best Practice – Use models
Step 2 – Design Targets

**Build**
- Owners/User Accounts
- Entities (maybe…)
- Relationships (maybe…)
- Datatypes (maybe…)
- Rules (maybe…)

Manually research conversions and create database and objects
Best Practice – Auto conversion with models (with a careful eye…)
Step 2 – Design Targets

A Few “Best Practices” From Successful Implementations

- Select storage engine based on need (MyISAM, InnoDB, Memory)
- If using referential integrity, enforce through MySQL or other DB platform
- Start with basic index strategy (PK, foreign key – no choice on InnoDB…)
- Verify data length matches (upcoming gotcha…)
- Version design changes
What’s freely available from the database vendors for data movement?

### Native Database Unload/Load Ability for Cross-Platform Access

<table>
<thead>
<tr>
<th></th>
<th>Unload</th>
<th>Load</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Oracle</strong></td>
<td>Unload from SQL*Plus</td>
<td>SQL*Loader</td>
</tr>
<tr>
<td><strong>Microsoft SQL Server</strong></td>
<td>BCP, DTS</td>
<td>BCP, DTS</td>
</tr>
<tr>
<td><strong>Sybase</strong></td>
<td>BCP</td>
<td>BCP</td>
</tr>
<tr>
<td><strong>DB2 LUW</strong></td>
<td>Export, Unload utility</td>
<td>Import, Load utility</td>
</tr>
<tr>
<td><strong>DB2 z/OS</strong></td>
<td>Unload Only of Reorg</td>
<td>Load utility</td>
</tr>
<tr>
<td><strong>MySQL</strong></td>
<td>mysqlldump, select into outfile</td>
<td>mysqlimport, load data infile</td>
</tr>
</tbody>
</table>
Step 3 - Design ETL Flows

SQL Server Example to MySQL - DTS

... Structures and Data
Step 3 - Design ETL Flows

Sybase to MySQL Example - BCP

... uses default tab delimited
A Few “Best Practices” From Successful Implementations

• Avoid knowledge silos (personnel).

• If using scripts/manual approach, version control migration scripts.

• Document everything (process flows, scripts, transformation routines, etc.)

• If using schedules or recurring ETL processes – use robust scheduler that supports job chaining, alerts, etc.

• Instrument ETL code (logging, etc.)

• Did you know that you can not only migrate data, but code also …?
Top MySQL ‘Gotchas’ to watch out for during migration…

- **Data integrity** – don’t depend on MySQL to enforce through datatype def’s.
- **Domains** – don’t depend on MySQL to enforce through `enum`.
- Watch for **rounding errors** on numeric datatypes with heavy precision.
- Watch for **data truncation** on numeric and character data values.
- **Duplicate data problems** – know the difference in storage engines (MyISAM lets duplicates through on mulit-record inserts; Innodb does not).
- Watch out for **multiple TIMESTAMP columns** in tables during updates.
- **NULL’s** – watch for multi-inserts with NULL’s (MySQL converts NULLs to default column value!)
A Few “Best Practices” From Successful Implementations

- Backup targets before migrations if not complete refresh (maybe even if…)
- Test load ‘rules of thumb’ beforehand.
- Only use the drop/recreate foreign key method if you’ve validated your data beforehand and are not loading in order of parent/child.
- Rebuild indexes, update stats, and optimize tables after heavy loads.
- Utilize advanced options for native utilities (direct load, multi-insert).
- Run data validation confirmation tests between source/target after completion.
- Preserve source data for agreed-upon period of time.
Step 4 - Build/Run Migrations

Myth or Fact: Drop all indexes prior to large load, then recreate...
Test of ~ 500,000 row load

Load with three indexes present - 27.16

Individual factors do contribute to overall times (type of data columns indexed, etc.)
Myth or Fact: Drop then recreate foreign key constraints on InnoDB…
Test of ~ 500,000 row load

Load with foreign key enabled – 63.56

Load then create foreign key – 127.7

Individual factors do contribute to overall times (type of data columns indexed, etc.)
Why do Performance Testing?

- Validates hardware, MySQL (or other DB) configuration.
- Pinpoints queries that ran fine on source but not on target.
- Provides confidence needed for 1st day of production.

Open source and 3rd party solutions available...
Step 5 – Performance test

A Few “Best Practices” From Successful Implementations

• First, optimize all structures and update statistics.
• Ensure representative user load (session simulation).
• Ensure representative data load.
• Save Benchmarks for performance comparisons.
Proven Database Migration Process

Data Migration Lifecycle

1. Document Sources
   - User Accounts
   - Entities
   - Relationships
   - Datatypes
   - Rules (data, etc.)

2. Design Targets
   - User accounts
   - Entities
   - Relationships
   - Datatypes
   - Rules (data, etc.)

3. Design ETL Flows
   - Simple moves
   - Data Transformations
   - Aggregations
   - Data Profiling/Cleansing

4. Build/Run Migrations
   - ETL Runs
   - Scheduled/Recurring
   - Validation
   - Migrate Code?

5. Performance Test
   - Simulation
   - Metric Analysis
   - Review

- Scripts/Manual
- Data Models
- Models, Reports, etc.
- Scripts/Manual
- Data Models
- Models, Reports, etc.
- Scripts/Vendor Supplied
- Tactical 3rd Party
- Enterprise Global
- Scripts/Manual
- Tactical 3rd Party
- Enterprise Global
- Manual
- Tactical 3rd Party
- Automated
Proven Database Migration Strategies

Thank You!

we make data work™