Data Warehousing with Perl
Colin Bradford
Data Warehousing with Perl

- An example operational schema
- Some typical reporting questions
- Answering with the operational database
- Introduction to Star schemas
- ETL : Extract, Transform and Load
- Answering with the Reporting database
- Things I have learnt
An example operational schema
Typical reporting questions

- How many customers have we got?
- How many discs have we shipped?
- How many customers did we have on this package last month?
- What's the 28day retention of this package?
- Which titles do we have too many discs for?
Answering with the operational database

- **How many customers have we got?**
  - SELECT COUNT(*) FROM customers WHERE status = 'Y'

- **How many discs have we shipped?**
  - SELECT COUNT(*) FROM dispatches WHERE date_shipped = NOW()

- **How many customers did we have on this package last month?**
  - Add transaction tables for package changes

- **What's the 28 day retention of this package?**
  - Add transaction tables to customers

- **Which titles do we have too many discs for?**
  - Probably pull the data into Perl to manipulate
Alternative: Reporting schema

- Take the operational data, and transform it
- Do this every night
- Time based series of data
- Much easier to report on
- Key aggregates already calculated
- Design the schema for ease of querying
  - Use a Star schema
Introduction to Star schemas

Fact tables
- Contain measurements – how many of this title are on the shelf
- Grain: eg one row per movie per day

Dimension tables
- Data element – for example, static information about a movie
- Shared across fact tables
- Can contain data from more than one operational table – title, genre, classification

Create surrogate keys for joins, not the operational PK/FK

Don't snowflake

Denormalise!
Example schema for movies
Example schema for customers
Tracking changes in dimensions

▶ **Type 1: Just update the dimension**
  - Loses historic data
  - Good for true facts, like movie title

▶ **Type 2: Create a new dimension row**
  - Keeps history, more data
  - Source Primary Key is not unique in the dimension
  - Allows tracking of changes
  - For example, price of a package

▶ **There are other methods**
  - But I haven't used them
Extract, Transform and Load

- Extract from source database
  - For example, take a snapshot
- Transform into data warehouse format
- Load into data warehouse
  - Separate step, because this step will slow the Data Warehouse
Transform: an example

- Update the movie_dim dimension and get a mapping from movie_id to movie_key
  - DBIc: update_or_create for type 1, find_or_create() for type 2
- Do calculations on source to get facts, indexed by movie_id
- Build movie_fact rows based on measures and keys
Answering with the Reporting database

- **How many customers have we got?**
  
  ```sql
  SELECT COUNT(*)
  FROM customer_fact
  WHERE status='ACTIVE'
  AND date = NOW()
  ```

- **How many discs have we shipped?**
  
  ```sql
  SELECT SUM(shipped)
  FROM movie_fact
  WHERE date=NOW()
  ```
Reporting: customer numbers

How many customers did we have on this package last month?

SELECT COUNT(*) FROM customer_fact
JOIN package_dim USING (package_key)
WHERE customer_fact.date = '2009-06-01'
AND package_dim.package_id = 23
What's the 28day retention of this package?

```
SELECT c2.STATUS, COUNT(*)
FROM customer_fact cf1
JOIN package_dim p ON p.package_key = cf1.package_key
JOIN customer_dim cd1 ON cf1.customer_key = cd1.customer_key
JOIN customer_dim2 cd2 ON cd2.customer_id = cd1.customer_id
JOIN customer_fact cf2 ON cd2.customer_key = cf2.customer_key
WHERE cf1.date = NOW() - INTERVAL 28 day
AND cf2.date = NOW()
AND p.package_id = ?
```
Reporting: disc usage

- **Which titles do we have too many discs for?**
  
  ```sql
  SELECT movie_id, COUNT(*) AS num_days
  FROM movie_fact
  JOIN movie_dimension USING (movie_key)
  WHERE discs_in_stock > 10
  AND date BETWEEN '2009-02-01' AND '2009-02-28'
  HAVING num_days > 27
  ```

- Scans 1 row per movie per day, rather than all shipments and returns that might cross that time period
Things I have learnt

- **Uses masses of disk space**
  - Eg, 1 row per customer (active and cancelled) per day, for a month, can be 100 million rows. 100 bytes a row = 10Gb/month growth

- **Instrument the ETL, to track individual steps for when it slows down**

- **Ensure you can rebuild a failed build**

- **Try and split the process into idempotent steps**
  - This makes rerunning a failed build easier
Thank you