Database and SQL (un)patterns

Frankfurt, November 5th 2007

International PHP Conference

Lukas Kahwe Smith (lsmith@optaros.com)

http://pooteeweet.org/files/phpconf07/sql_un_patterns.pdf

optaros

Who am I?

- PHP since PHP3 days
- Joined PEAR late 2002
- Spamming internals@ mailinglist since 2003
- Unofficial todo wiki maintainer
- Employed at Optaros
  - Doing LAMP based NGI applications
  - NGI stands for Next Generation Internet aka Web 2.0
  - We (and pretty much all of Switzerland) are hiring!

optaros

Programming is nice, but ..

optaros

Would have love to have caught that one ..

So it goes

optaros

This is how I spend this weekend ..

optaros

Anyways, today's most important message ..

optaros

Speakers will try to entice you to listen and think hard ..

optaros

But conferences are about socializing ..

optaros

Do not go home without having talked to at least 10 people face to face for a minimum of 10 minutes!

optaros

Ok! Enough of the one sentence per slide nonsense. Lets get to the meat!

optaros

A couple of things upfront

- Feel free to interrupt
- Talk will cover a lot of ground
- Not enough time to go into detail on all topics
- Aim is to familiarize with the concept and terms
  - Wikipedia, Google, Yahoo etc. are your friends!

optaros

Grand agenda for today

- Schema Design
- Indexes and Constraints
- SELECT magic
- Data changes
- Unusual data

optaros

Schema Design

- Data Types
- Normalization
- Surrogate Keys
- DOMAIN and ENUM
- EAV Pattern
- Trees
- Queues

optaros

Data Types

- **Take a moment to think about VARCHAR length**
  - Sort buffers use the maximal length for variable length columns
    - Do not pick largest possible length, better think about what happens if something is too large
    - Maybe a CLOB (aka TEXT) is the right choice?
- **Choose by what functions will be applied on the column**
  - Phone numbers will often require string functions
  - Unix timestamp vs ISO timestamp
- **Minimize type casts**

optaros

Normalization Overview

- **Hierarchical rules for removing redundant data from tables**
  - Helps avoiding INSERT, UPDATE and DELETE anomalies
- **Multiple normal forms (NF)**
  - Aim for 3NF by default
  - Beyond that it gets somewhat obscure and rarely relevant for real world applications
- **Denormalize to fix perf. issues**
  - Balance slow down for INSERT/UPDATE/DELETE with performance improvements for SELECT
  - Requires additional logic to manage redundant data

optaros

Normalization Example Intro

Example: Name PK, Salary CK

- PRIMARY KEY (PK) uniquely identifies a specific row
- CANDIDATE KEY (CK) is any group of columns that uniquely identify rows of a data set

Name	Title	Language	Salary	Workgrou	Head
Axworthy	Consul	French,Germa	30,000	WHO,IMF	Greene,Craig
Broadbent	Diplomat	Russian,Greek	25,000	IMF,FTA	Craig,Candall
Craig	Amabassador	Greek,Russian	65,000	IMF	Craig
Candall	Amabassador	French	55,000	FTA	Candall
Greene	Amabassador	Spanish,Italian	70,000	WHO	Greene

optaros

Normalization First Normal Form (1NF)

- **All columns only contain scalar values (not lists of values)**
  - Split Language, Workgroup, Head
  - Name, Language and Workgroup are now the PK
- **Add all possible permutations?**

Name	Title	**Language**	Salary	**Workgroup**	Head
Axworthy	Consul	French	30,000	WHO	Greene
Axworthy	Consul	German	30,000	IMF	Craig
Broadbent	Diplomat	Russian	25,000	IMF	Craig
Broadbent	Diplomat	Greek	25,000	FTA	Candall
Craig	Amabassador	Greek	65,000	IMF	Craig
Craig	Amabassador	Russian	65,000	IMF	Craig
Candall	Amabassador	French	55,000	FTA	Candall
Greene	Amabassador	Spanish	70,000	WHO	Greene
Greene	Amabassador	Italian	70,000	WHO	Greene

optaros

Normalization Dependence

- **A column is**
  - **Set dependent** if its values are limited by another column
  - **Functionally dependent** if for every possible value in the column, there is one and only one possible value set for the items in a second column
    - Must hold true or all possible values
  - **Transitively dependent** on another column C if that column is dependent on column B which in turn is dependent on column C

optaros

Normalization Second Normal Form (2NF)

- **All non-key columns must be functionally dependent on PK**
  - Title, Salary are not functionally dependent on the Language column
  - Head is set dependent on Workgroup

Name	**Language**
Axworthy	French
Axworthy	German
Broadbent	Russian
Broadbent	Greek
Craig	Greek
Craig	Russian
Candall	French
Greene	Spanish
Greene	Italian

Name	Title	Salary	**Workgroup**	Head
Axworthy	Consul	30000	WHO	Greene
Axworthy	Consul	30000	IMF	Craig
Broadbent	Diplomat	25000	IMF	Craig
Broadbent	Diplomat	25000	FTA	Candall
Craig	Amabassa	65000	IMF	Craig
Candall	Amabassa	55000	FTA	Candall
Greene	Amabassa	70000	WHO	Greene

optaros

Normalization Third Normal Form (3NF)

- **All non-key columns must be directly dependent on the PK**
  - Head is only dependent on the Name through the Workgroup column

Name	**Language**
Axworthy	French
Axworthy	German
Broadbent	Russian
Broadbent	Greek
Craig	Greek
Craig	Russian
Candall	French
Greene	Spanish
Greene	Italian

Name	Title	Salary
Axworthy	Consul	30000
Broadbent	Diplomat	25000
Craig	Amabassador	65000
Candall	Amabassador	55000
Greene	Amabassador	70000

Name	**Workgrou**
Axworthy	WHO
Axworthy	IMF
Broadbent	IMF
Broadbent	FTA

Workgroup	Head
FTA	Candall
IMF	Craig
FTA	Candall
WHO	Greene

optaros

Surrogate Keys

- **Natural key (NK) is a CK with a logical relationship to that row**
- **Surrogate key (SK) is an artificially added unique id**
  - A lot of ORM's, AR's and Martin Fowler love SKs
  - Since they are artificial they make query logs hard to read and can lead to more JOINs
    - SELECT city.code AS citycode, country.code AS countrycode FROM city, country WHERE city.country_id = country.id AND city.country = 'DE'
  - Integers do not significantly improve JOIN performance or reduce file I/O for many data sets
  - Can help in making sure the PK is really immutable

optaros

DOMAIN and ENUM

- **Help avoid using lookup tables to model static constraints**
  - DOMAIN is a data type with optional constraints
  - ENUM allows for limiting the possible string values
    - Also allows custom sorting and compact storage
  - Much faster than JOIN on look up table
  - More obvious when reviewing the schema
- **Changing a DOMAIN/ENUM requires DDL!**
  - For MySQL this means the entire table needs to be rebuild via an ALTER TABLE command

optaros

Entity Attribute Value Pattern

- **EAV is using type, name, value columns to store anything**
  - Value usually has to be set to a very large VARCHAR
  - No way to model uniqueness or other constraints on values efficiently
  - Unrelated data is stored in the same table
- **Alternatives**
  - Look for a proper relational structure
    - If necessary generate DDL on the fly
    - Use sub selects and UNION's to relate the separate tables
  - Work around number of column limitations with splitting table into a set of hardcoded 1:1 tables

optaros

Adjacency model

Text book approach

- Each row stores the id to its parent
- Root nodes have no parent
- Self JOINs are needed to read more than one depth level in a single query
- Depth levels to read are hardcoded into the query
  - SELECT t1.name name1, t2.name name2, t3.name3 FROM tbl AS t1 LEFT JOIN tbl AS t2 ON t2.parent = t1.id LEFT JOIN tbl AS t3 ON t3.parent = t2.id WHERE t1.name ` foo';
- Sub tree can be moved by modifying a single row

id	parent_id	name
1	NULL	US HQ
2	1	Europe
3	2	Switzerland
4	2	Germany

optaros

Materialized Path

- **Reference parent PK through the full path for each child**
  - Violation of normalization rules
  - No JOIN necessary to fetch entire tree as well as vertical or horizontal sub tree's
    - SELECT * FROM tbl ORDER BY path, name
    - SELECT * FROM tbl WHERE path LIKE '/1/23/42/%' ORDER BY path, name
    - SELECT * FROM tbl WHERE path LIKE '/1/_' ORDER BY name
      - Optionally also store a depth column to get rid of the LIKE
  - Moving subtrees only requires changes to path column for all rows in the subtree
    - UPDATE tbl SET path = CONCAT('/1/5/19', SUBSTRING(path, LENGTH('/1/23/42')) WHERE LIKE '/1/23/42%'
  - Need to know node path

Nested Sets

- **Store left and right node number for each row**
  - Start counting up from one left of the root node while moving around the outer edges
  - Very fast read performance
  - Very slow write performance

Personnel emp	lft	rgt
'Albert'	1	12
'Bert'	2	3
'Chuck'	4	11
'Donna'	5	6
'Eddie'	7	8
'Fred'	9	10

Table 2: Nested Set Model

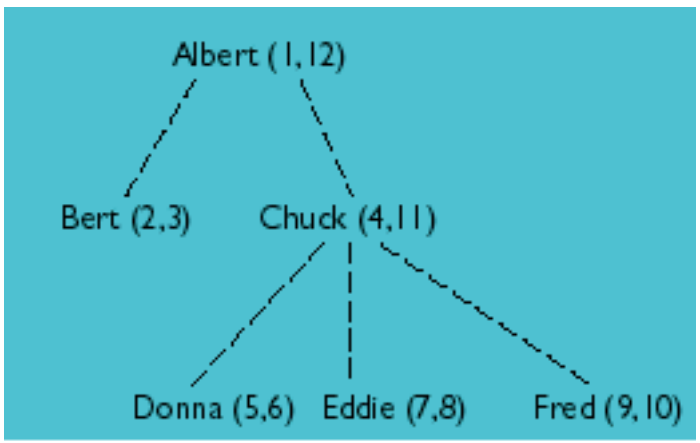

Figure 1: Directed graph.

optaros

Nested Sets

- **Some example queries**
  - Get the entire path to Donna
    - SELECT * FROM pers WHERE lft <= 5 AND rgt >= 6
  - Get all leaf nodes
    - SELECT * FROM pers WHERE rgt - lft = 1
  - Get subtree starting attached to Chuck
    - SELECT * FROM pers WHERE lft > 4 AND rgt < 11
- **Changes to the tree require updating a lot of rows**
  - Need to know left and right node number
  - Cannot be hand maintained
  - Results in meaningless numbers inside queries when examining log files

optaros

Fixed Length Queues

- **Sample implementation of a fixed length queue**
  - INSERT INTO q (id, modulo, fruit) SELECT (COALESCE(MAX(id), -1) + 1), (COALESCE(MAX(id), -1) + 1) MOD 5, :fruit FROM q ON DUPLICATE KEY UPDATE id = values(id), fruit = values(fruit);
  - REPLACE INTO q (id, modulo, fruit) SELECT (COALESCE(MAX(id), - 1) + 1), (COALESCE(MAX(id), - 1) + 1) mod 5, :fruit FROM q;
  - CREATE RULE fruitlimit AS ON INSERT TO q DO ALSO DELETE FROM q WHERE id NOT IN (SELECT id FROM q ORDER BY DESC LIMIT 5);
  - Alternatively use the SQL standard MERGE

optaros

Indexes and Constraints

- Over indexing
- Covering indexes
- Full text indexes
- FOREIGN KEY
- Emulating CHECK constraints

optaros

Over indexing

- Any index needs to be updated when index columns change
  - Indexes slow most INSERTs, UPDATEs and DELETEs
- Legacy DBMS don't support on the fly index merging
  - Even if they do in most cases DBMS can only use one index per table efficiently
- Use compound indexes instead
  - Index on (a, b, c) implies an index on (a) and (a, b)
- Make sure indexes are used!

optaros

Covering indexes

- **Skip reading table rows when all relevant data is inside index**
  - Table foo has compound index on column a, b and c
  - SELECT b, c FROM foo WHERE a IN (..)
  - Consider adding select list columns into PK indexes
  - Make sure you use the right order in the index!
    - For this case only (a, b, c) or (a, c, b) would work
    - Watch out for write overhead if b and c are often changed
  - Very useful in tables that also have rarely read LOBs when using DBMS that stores LOBs inline (MySQL)
  - Indexes are like DBMS managed denormalization

optaros

Full Text Indexes

- **Add search engine like functionality to DBMS**
  - LIKE '%foo%' and even LIKE '%foo' cannot use index
  - Boolean operators supported
    - SELECT * FROM articles WHERE MATCH(title, body) AGAINST ('+MySQL -YourSQL' IN BOOLEAN MODE);
  - Row relevance supported
    - SELECT id, MATCH (title, body) AGAINST ('Tutorial') FROM articles
  - Much faster than a normal index on large data sets
  - Only supported for MyISAM on MySQL
    - Replicate data into a MyISAM table via TRIGGER to combine InnoDB features with FULL TEXT INDEX support
  - Different syntax on each DBMS

optaros

FOREIGN KEY

- **DBMS manages integrity of relations with FOREIGN KEYs**
  - Ensure that parent row exists in lookup table
    - FOREIGN KEY (parent_id) REFERENCES lookuptbl (id)
  - Automatically act on child row when parent row is updated or deleted
    - ON UPDATE CASCADE
    - ON DELETE RESTRICT
    - ON DELETE SET NULL
  - Much safer than having an ORM or worse hand maintained code handle this
    - Also works when working via CLI or some query browser
  - Only supported for InnoDB on MySQL
    - MySQL's Falcon Engine will not support FKs until 6.1!

optaros

Emulating CHECK constraints

- **Missing support for CHECK constraints in MySQL**
  - Use VIEW and "WITH CHECK OPTION"
    - This option prevents any INSERT/UPDATE that will cause the row to not match underlying VIEW SELECT
  - CREATE VIEW t_with_check AS SELECT * FROM t WHERE some_col < 20;
  - INSERT INTO t_with_check VALUES (2); // OK
  - INSERT INTO t_with_check VALUES (23); // FAIL
  - Use ENUM and SET to limit allowed string values

optaros

Timeout!

optaros

SELECT magic

- SELECT *
- Optimizating
- Case for CASE
- ORDER BY RANDOM()
- GROUP BY
- Pivot tables
- Ranking

optaros

SELECT *

- What columns are read?
- May break test passing code
  - A new column is added to an application
  - Developer decides to add it in a specific place in the full SQL dump but users adds it at the end of table
- Useful CLI (and examples)
- Do not use it in production

optaros

Premature Optimization

- **Using surrogate keys or denormalization without**
  - Seeing a real world specific bottleneck to fix
  - Understanding of what will be slowed down as a result
- **Using fancy non standard features unless necessary**
  - Do not think too hard on this one
- **Thinking too much about future scalability problems**

optaros

Forgetting About Optimization

- **Testing performance on unrealistic data**
  - Test on expected data size
  - Test on expected data distribution
  - Most benchmark tools offer data generators
- **Not thinking about scalability requirements beforehand**
  - This one is a fine balance and gets easier with experience
  - Do not be afraid to draw upon outside experts if the expectation is to grow up quick

optaros

Case for CASE

- **Cut down on function calls**
  - WHERE some_slow_func() = 'foo' OR some_slow_func() = 'bar'
  - WHERE CASE some_slow_func() WHEN 'foo' THEN 1 WHEN 'bar' THEN 1 END
- **Fold multiple queries into one**
  - foreach ($rows as $id => $row)
    - if (..) UPDATE foo SET r * 0.90 WHERE id = $id
    - else UPDATE foo SET r * 1.10 WHERE id = $id
  - UPDATE foo SET r = (CASE WHEN r > 2 THEN r * 0.90 ELSE r * 1.10 END);

optaros

ORDER BY RANDOM()

- **ORDER BY RANDOM()**
  - Obvious but slow
- **WHERE id = :random_id**
  - random_id = rand(1, @max)
  - Only works if MAX(id) == COUNT(id)
- **WHERE id >= :random_id ORDER BY id LIMIT 1**
  - Not truly random when distribution is not even
    - JOIN with a derived (on the fly or denormalized) table with additional rank column

optaros

Non deterministic GROUP BY

- **Columns in select list need to be either inside an aggregate the GROUP BY clause**
  - SELECT id, addr, MAX(age) FROM t GROUP BY id
    - Will be deterministic only of id is a PK or CK for addr
  - SELECT name, addr FROM t GROUP BY name
    - Will be non deterministic if there are multiple different addr values for a given name
  - Allowed in MySQL
    - Disable with "ONLY_FULL_GROUP_BY" SQL mode
  - Rewrite SQL statement to be deterministic
    - SELECT id, MIN(addr), MAX(age) FROM t GROUP BY id
    - SELECT name, addr FROM t GROUP BY name, addr

optaros

More GROUP BY Functions and Modifiers

- **Get all values in a group with MySQL's GROUP_CONCAT()**
  - SELECT continent, GROUP_CONCAT(country) FROM country GROUP BY continent
- **Add extra super aggregates on all levels WITH ROLLUP**
  - SELECT id, SUM(bar) FROM t GROUP BY id WITH ROLLUP

id	SUM(bar)
23	11132
42	20205
NULL	31337

optaros

Group-wise Max

- **Find country names with highest population for each continent**
  - SELECT name, pop FROM country c1 WHERE pop = (SELECT MAX(c2.pop) FROM country c2 WHERE c1.continent = c2.continent)
    - Solve execution of sub select once per row with a temp table
  - SELECT name, pop FROM country WHERE ROW(pop, continent) IN (SELECT MAX(c2.pop), continent FROM country GROUP BY continent)
  - SELECT SUBSTRING(MAX(CONCAT(LPAD(pop, 10, '0'),name)), 10+1) AS name, MAX(pop) AS pop FROM country GROUP BY continent

optaros

Pivot tables

- **Statistical reports often call or grouping data by one field**
  - Also known as cross-tabs or breakdown
  - SELECT name,
    COUNT(CASE WHEN gender = 'm' THEN id ELSE NULL END) AS 'males',
    COUNT(CASE WHEN gender = 'f' THEN id ELSE NULL END) AS 'females',
    COUNT(*) total FROM person GROUP BY department
  - Challenges to over come
    - Having to derive data from multiple tables
    - Cross between more than one value horizontally/vertically
    - More values will require more subtotals
    - Generalization to handle other cases with the same code

optaros

Pivot Tables

Adding more dimensions

- **Consider country and location**

```
SELECT country, loc AS location,
   COUNT(CASE WHEN dept = 'pers' AND gender = 'f' THEN id ELSE NULL END) AS 'pers-f',
   COUNT(CASE WHEN dept = 'pers' AND gender = 'm' THEN id ELSE NULL END) AS 'pers-m',
   COUNT(CASE WHEN dept = 'pers' THEN id ELSE NULL END) AS 'pers',
   COUNT(CASE WHEN dept = 'sales' AND gender = 'f' THEN id ELSE NULL END) AS 'sales-f',
   COUNT(CASE WHEN dept = 'sales' AND gender = 'm' THEN id ELSE NULL END) AS 'sales-m',
   COUNT(CASE WHEN dept = 'sales' THEN id ELSE NULL END) AS 'sales',
   COUNT(CASE WHEN dept = 'dev' AND gender = 'f' THEN id ELSE NULL END) AS 'dev-f',
   COUNT(CASE WHEN dept = 'dev' AND gender = 'm' THEN id ELSE NULL END) AS 'dev-m',
   COUNT(CASE WHEN dept = 'dev'  THEN id ELSE NULL END) AS 'dev',
   COUNT(*) AS total
FROM person
INNER JOIN depts ON (person.dept_id=depts.dept_id)
INNER JOIN locs ON (locs.loc_id=person.loc_id)
INNER JOIN countries ON (locs.country_id=countries.country_id)
GROUP BY country, loc
```

optaros

Ranking

- **SQL 99 "windowing functions"**
  - Supported in Oracle, DB2 (hopefully PostgreSQL 8.4)
  - SELECT * FROM (SELECT RANK() OVER (ORDER BY age ASC) AS ranking, person_id, person_name, age FROM person) AS foo WHERE ranking <= 3
    - Find the people with the 3 lowest ages (including ties)
- **Alternatively use self JOIN**
  - SELECT * FROM person AS px WHERE (SELECT COUNT(*) FROM person AS py WHERE py.age < px.age) < 3
- **Find rank of a user by score**
  - SELECT COUNT(*)+1 AS rank FROM points WHERE score > (SELECT score FROM points WHERE id = :id)

optaros

Ranking MySQL style

- **Find highest five salaries for each department**
  - SELECT dep, sal, rank FROM
    (SELECT dep, sal,
    CASE WHEN @D = dep
    THEN @R:=@R=1 ELSE @R:= 1 END rank,
    CASE WHEN @D != dep
    THEN @D:=dep END
    AS g FROM tbl ORDER BY dep, sal) AS tbl
    WHERE rank <= 5

optaros

Data Changes

- SQL Injection
- Optimistic Locking
- Affected Rows
- Overwriting
- MVCC
- Smart transactions
- Surrogate key generation
- CSV import/export

optaros

SQL Injection

Always quote!

- Quote, validate, filter all data and identifiers from external resources
  - Who says you can trust any external source?
  - What happens when you refactor?
- Do not rely on magic_quotes_gpc or addslash()

Quote context aware

- mysqli_real_escape_string() with connection link!
- PDO::quote(), MDB2::quoteIdentifier()
- Prepared statements
  - SELECT id, name FROM foo WHERE bar = :bar
  - SELECT id, name FROM foo WHERE bar = ?
- Hardcoded whitelist
  - switch ($foo) {case 'bar': $param = 'bar'; default: die();}

optaros

Optimistic Locking

Locking at COMMIT time

- Assume that no other transaction modifies the data between independent read and write transactions
- Fail instead of wait on concurrent transactions
- Allows using isolation level READ UNCOMMITTED
- Requires adding a unique "counter" to the PK
- INSERT INTO addr (id, cntr, street, city) VALUES (nextval('addr')), unix_t(), 'Foo Street', 'Bar Town');
  - id - 23, unix_t() = 1179145598
- SELECT id, cntr, street, city FROM addr;
- UPDATE addr SET street = 'Foo Ave', cntr = unix_t() WHERE id = 23 AND cntr = 1179145598;
- Check affected rows
  - affected rows is 1 => Success
  - affected rows is 0 => Failure

optaros

Affected Rows

- **Check affected rows to avoid SELECT before data change**
  - Check if affected rows after UPDATE/DELETE > 0
    - Something was updated/deleted
- **Good ORM supports UPDATE/DELETE without SELECT**
  - $dql->create()->delete()->from('foo')->where('bar = :bar', array('bar', 42))->execute();

optaros

Overwriting

Overwriting any existing rows

- MySQL/SQLite REPLACE is generally a bad idea
- MySQL's INSERT INGORE to ignore dupe violations
- MySQL's INSERT .. ON DUPLICATE KEY UPDATE
  - Great to handle multiple counters per table
    - Though in MyISAM one can also use multi column auto increment for this
- Custom stored routine

Ensure that a row exists in a table

- MySQL's INSERT INGORE to ignore dupe violations
- Custom stored routine

optaros

MVCC Problems

MVCC prevents readers from getting blocked

- Readers get a snapshot of the data which was valid at the start of the reader transaction
- Can lead to issues with concurrent transactions

Transaction #1	Transaction #2	Comments
BEGIN TRANS;		
	BEGIN TRANS;	
SELECT FLIGHT 23		Seat 1A available
UPDATE FLIGHT 23		Book 1A
	SELECT FLIGHT 23	Seat 1A available
COMMIT		
	UPDATE FLIGHT 23	Book 1A
	COMMIT	

optaros

MVCC Solutions

- **Add check in UPDATE**
  - SET customer = ‘foo’ WHERE flight = ‘23’ AND seat = ‘1A’ AND **customer IS NULL**;
  - Look at affected rows to check for concurrent writes
- **Use FOR UPDATE to acquire a lock in transaction**
  - SELECT seat FROM seats WHERE flight = ‘23’ AND customer IS NULL **FOR UPDATE**
  - Disables benefits MVCC for the SELECT

optaros

Smart transactions

- **Most DBMS do not support nested transactions**
  - Without nested transactions its hard to make effective use of transactions in modular applications
  - Use SAVEPOINTs to emulate nested transactions
    - $dbh->beginNestedTransaction(); # BEGIN TRANS
    - call_module($dbh)
      - $dhh->beginNestedTransaction(); # SET SAVEPOINT 1;
      - if ($fail)
        - » $dbh->failNestedTransaction(false); # set nested transaction as failed
      - else
        - » $dbh->completeNestedTransaction(); # RELEASE SAVEPOINT 1;
    - $dbh->completeNestedTransaction(); # COMMIT/ROLLBACK;
    - if ($dbh->getNestedTransactionError())

optaros

Surrogate key generation

- **Sequences vs Autoincrement**
  - Sequences can be used for any unique key generation including dynamic table names
    - Autoincrement can emulate sequences
  - Make sure to initialize generators when deploying
- **Don't clean SK "holes"**
  - Point of a surrogate key is to ensure uniqueness not that they are sequential or in any particular order
- **Alternative generators**
  - UUID()
  - Timestamp
    - Watch out for multiple concurrent INSERTs per millisecond

CSV import/export

- **Most RDBMS have some syntax to quickly import/export CSV**
  - MySQL: LOAD DATA INFILE
  - PostgreSQL: COPY
- **For multiple INSERTs bulk imports disable constraints**
  - MySQL: ALTER TABLE [ENABLE|DISABLE] KEYS
- **Use the MySQL CSV storage engine as interface to CSV files**

optaros

Unusual data

- Complex Data Structures
- Images in the database

optaros

Complex Data Structures

- **Same data structures are inefficient to normalize**
  - Configurations that can have an arbitrary structure
  - Large number of optional data fields require EAV
- **Use XML**
  - If data is sometimes queried
  - If structure/data needs to be validated
- **Use serialized strings**
  - If there is no intend to ever query inside the data
    - Make sure data does not better fit inside the code or configuration file that can be managed inside an SCM

optaros

Images in the database

- **Several good reasons for storing LOBs in an DBMS**
  - Leverage DBMS replication
  - Leverage DBMS backup
  - Leverage DBMS access control
  - Leverage DBMS transactions
  - Leverage DBMS OS portability
  - Use mod_rewrite to cache public images in the FS
    - mod_rewrite points missing images to a script with the name as a parameter
    - script pulls out the image from the database
      - if the image is public its cached in the FS
    - script returns image
  - MyBS brings http based LOB streaming to MySQL

optaros

References

- MySQL and PostgreSQL online documentation
- SQL Performance Tuning by Peter Gulutzan and Trudy Plazer
- http://jan.kneschke.de/projects/mysql
- http://decipherinfosys.wordpress.com/2007/01/29/name-value-pair-design/
- http://www.xaprb.com/blog/2007/01/11/how-to-implement-a-queue-in-sql/
- http://people.planetpostgresql.org/greg/index.php?/archives/89-Implementing-a-queue-in-SQL-Postgres-version.html
- http://www.onlamp.com/pub/a/onlamp/2003/12/04/crosstabs.html
- http://arjen-lentz.livejournal.com/56292.html
- http://forums.mysql.com/read.php?32,65494,89649#msg-89649
- http://troels.arvin.dk/db/rdbms/
- http://www.intelligententerprise.com/001020/celko.jhtml?_requestid=1266295
- http://archives.postgresql.org/pgsql-hackers/2006-01/msg00414.php
- http://www.nexen.net/images/stories/conferences/mysqlkitchen.mce.pdf.zip
- Suggested topics by Robert Treat and links provided by the #postgresql channel bot
- My own blog http://pooteeweet.org and previous talks linked on the site