Bayerisches Staatsministerium für
Ernährung, Landwirtschaft und Forsten

PostgreSQL im Bayerischen Staatsministerium für Ernährung, Landwirtschaft und Forsten

08.11.2013

Michael Gengenbach

BayStMELF, Referat P5

Agenda

- Aufgaben
- Systeme
- Migration
- Erfahrungen

Integriertes Verwaltungs- und Kontrollsystem (InVeKoS)

- Durchführung der Agrarförderung gemäß Landes-, Bundes- und EU-Recht

Mehrfachantrag

Antragsteller (Name, Vorname bzw. Unternehmensbezeichnung): Huber, Josef, Testbetrieb MFA
Geburtsdatum* (TT/MM/JJJJ): 01.01.1960
BY Betriebsnummer: 09 910 100 2834
Straße, Hs.-Nr, Ortsteil: Ludwigstr. 2
PLZ, Ort: 80535 München
Bank (Name, Ort): BayernLB München
Bankleitzahl: 70050000 Kontonummer: 12345678
IBAN: BIC: BYLADEMMXXX
Telefon: 089/2182-0 Fax: 089/218227' Mobil-Tel.: 0123-4567890
E-Mail-Adresse: Josef.Huber@stmelf.bayern.de
* Gründungsdatum bei Personengemeinschaften bzw. juristischen Personen

An das
Amt für Ernährung, Landwirtschaft und Forsten

Test
Teststraße 11
99999 Testort 910

Mehrfachantrag 2013

Ich beantrage hiermit:

- Betriebsprämie durch Aktivierung der Zal
- Ausgleichszulage in benachteiligten Gebi
- Auszahlung 2013 für: 04
  - Kulturlandschaftsprogramm (KULAP)
  - Vertragsnaturschutzprogramm (VNP/E

Anlagen

- Flächen- und Nutzungsnachweis (FNN) mit Betrieb
- Viehverzeichnis (Anlage 2)
- Auszug aus der Digitalen Feldstückskarte (FeKa)
- gepachtete Zahlungsansprüche (Anlage Pacht-ZA
- Mitteilung Betriebsinhaberwechsel/betriebliche Ve
- Angaben zur KULAP-Maßnahme „Sommerweide

- Die Angaben in diesem Antrag und seinen Anlagen im Jahr 2013.
- Vor dem Ausfüllen bitte **Broschüre „Cross Compl**
- Der Antrag kann nur bearbeitet werden, wenn die A und er rechtzeitig, **spätestens bis zum Antragsen** Forsten eingereicht wird. Verspätet eingegangene A

① **Für Antragsteller, die ausschließlich am VNP/E B 4 und C maßgeblich, einschließlich Flächen-**

StMELF – P2/352-01.2013

ANLAGE 1 zum Mehrfachantrag 2013

Betriebs-Nr.: 910 100 2834

Betrieb: Testbetrieb MFA Josef Huber, Ludwigstr. 2, 80535 München
Feldstücksfläche gesamt: 9,14 ha

1	2	3	4	5	6	7	8	9
Gemark. Nr.	Flurstück	Zugeteilte Fläche	Nutzung 2012 mit LE				S	Code
			S	Code	ha, ar	B, N		

Feldstück: 2 Rennwegacker Reisen

FID: DEBYLI 8363001437			Gebiet: 5				Gesamt: 1,26 ha; CC-Erosionsklassen: Wa	
8363	521	0,0002	1	171	1,26	B	1	
8363	554	0,0005						
8363	555	1,2608						

Feldstück: 3 Reisener Feld

FID: DEBYLI 8363001448			Gebiet: 5				Gesamt: 1,02 ha; CC-Erosionsklassen: Wa	
8363	585	0,0190	1	411	1,02	B	1	
8363	644/1	1,0022						

Feldstück: 4 Inn.Sieben Tagwerk

FID: DEBYLI 8363001460			Gebiet: 5				Gesamt: 0,34 ha; CC-Erosionsklassen: Wa	
8363	1970/2	0,3394	1	560	0,34	B	1	

Ökoflächenkataster (0,34 ha)

Feldstück: 6 Dammacker Groll

FID: DEBYLI 8363001494			Gebiet: 5	Gesamt: 4,05 ha; CC-Erosionsklassen: Wa	
8363	442	0,0006	Keine Angaben, da Feldstück geändert	1	
8363	580	0,0001			
8363	614	1,1411			
8363	615	1,3409			
8363	616	1,5644			

Erläuterung: Siehe Anleitung zum Ausfüllen des FNN 2013 - blaue Farbe, S = Schlag, CC-

Feldstück				Fläche in ha, ar		
Nr.	Name	Flächenidentifikator (FID)	S	LF	AUM	LE
102	Stiftlacker	DEBYLI-8156-000171		1,30		

Maßstab (ca.): 1 : 2500

Feldstücksgrenze:
Schlaggrenze:
LE-Grenze:
Flurstücksgrenze:

Der Jahreszyklus der Agrarförderung

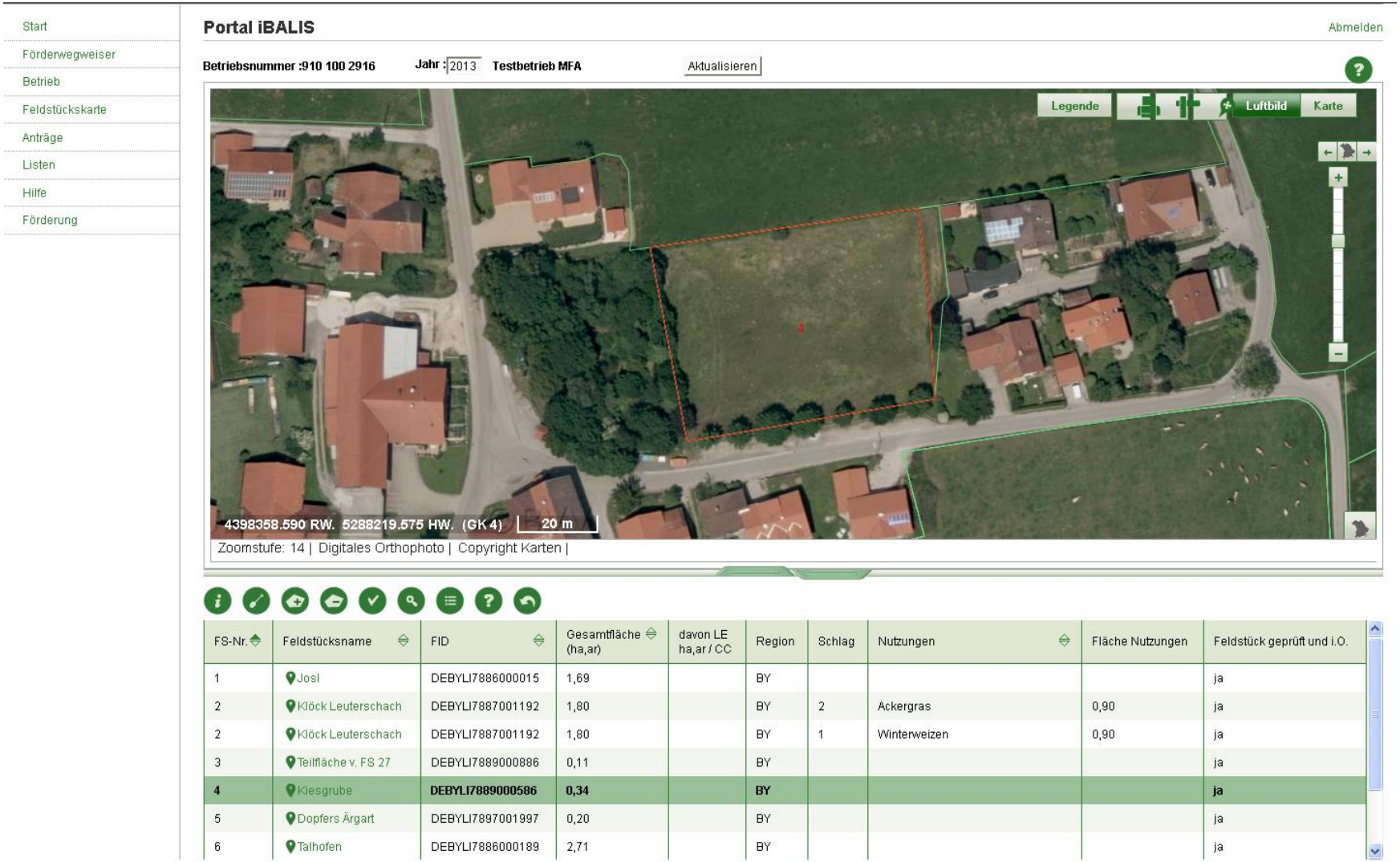
- Pflege der Feldstückskarte
- Generierung personalisierter Antragsformulare (als PDF)
- Erfassung der Anträge (online und über Papier)
- Verwaltungskontrolle
- Vor-Ort-Kontrolle nach Risikoanalyse
- Berechnung und Auszahlung der Förderbeträge
- Bereitstellung von Statistik- und Monitoring-Daten

- Rahmenbedingung:
  - Auch ältere Antragsjahre müssen berechenbar sein

Geodaten in der Agrarförderung

- Eigene Daten
  - Feldstücke
  - Landschaftselemente
  - …
- Geobasisdaten
  - Orthofotos
  - Verwaltungsgrenzen
  - Flurstücke
  - …
- Gebietskulissen
  - Wasserschutzgebiete
  - Naturschutzgebiete
  - …

Die Portalanwendung iBALIS

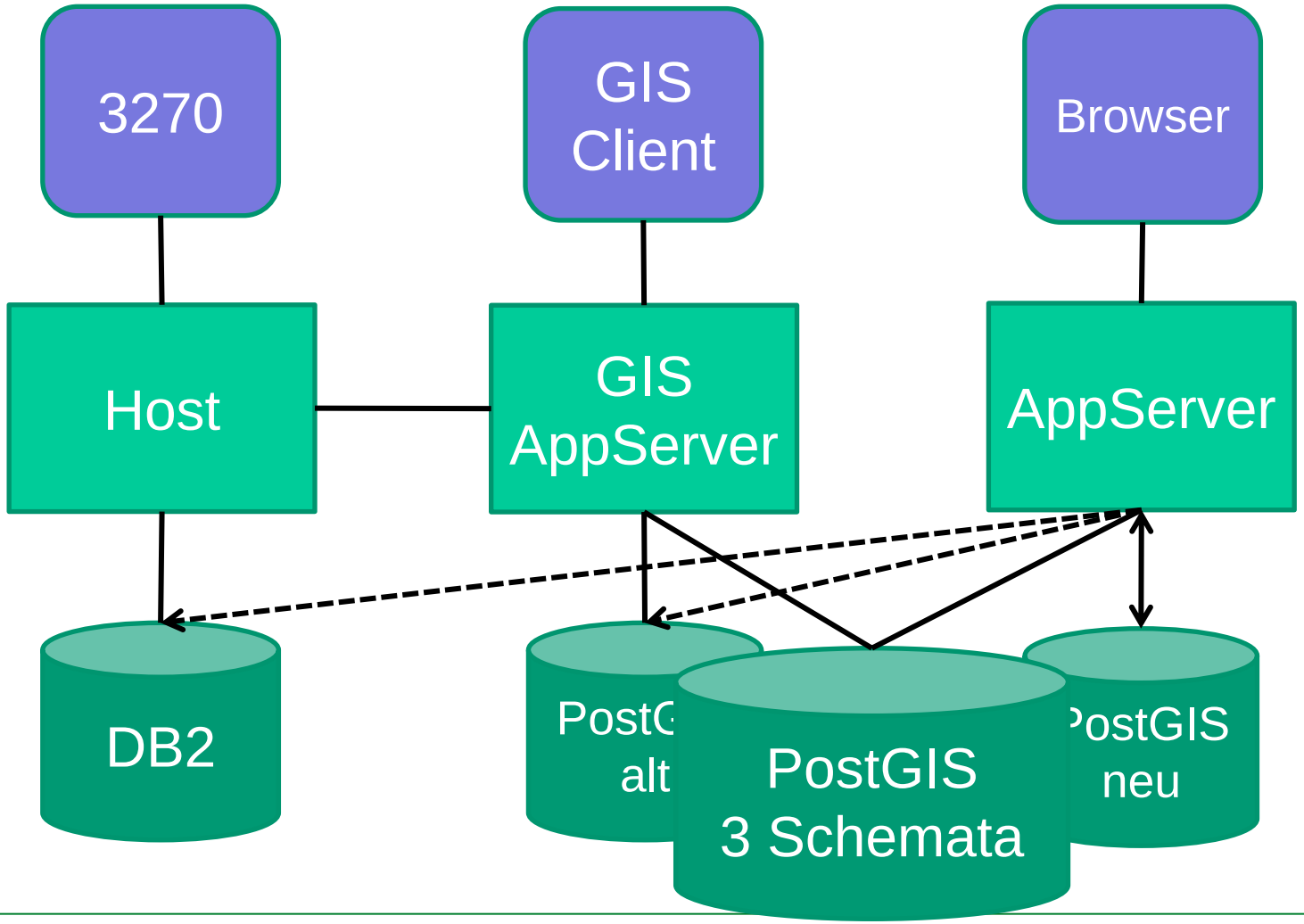


FS-Nr.	Feldstücksname	FID	Gesamtfläche (ha,ar)	davon LE ha,ar / CC	Region	Schlag	Nutzungen	Fläche Nutzungen	Feldstück geprüft und i.O.
1	Josl	DEBYLI7886000015	1,69		BY				ja
2	Klöck Leuterschach	DEBYLI7887001192	1,80		BY	2	Ackergras	0,90	ja
2	Klöck Leuterschach	DEBYLI7887001192	1,80		BY	1	Winterweizen	0,90	ja
3	Teilfläche v. FS 27	DEBYLI7889000886	0,11		BY				ja
4	Kiesgrube	**DEBYLI7889000586**	**0,34**		**BY**				**ja**
5	Dopfers Ärgart	DEBYLI7897001997	0,20		BY				ja
6	Talhofen	DEBYLI7886000189	2,71		BY				ja

Unser Weg zu PostgreSQL

2004	PostGIS für Geodaten (2 Mio. Feldstücke, 10 Mio. Flurstücke pro Jahr)
2011	PostgreSQL für kleinere neue Anwendungen
2012	Modernisierung der InVeKoS-Flächenverwaltung mit einem integriertem Datenmodell für GIS- und Sachdaten
2013	Beginn der Migration weiterer InVeKoS-Anwendungen auf das integrierte Datenmodell
ab 2014	Automatisierte Konvertierung aller Natural-InVeKoS-Anwendungen nach Java und Migration aller Daten nach PostgreSQL

Systeme


3270
GIS Client
Browser
Host
GIS AppServer
AppServer
DB2
PostG alt
PostGIS 3 Schemata
PostGIS neu

Größenordnungen der PostgreSQL-Datenbank

- Daten
  - 300 GB
  - 300 Tabellen
  - 400 Mio. Datensätze (viele mit Geodaten)
  - 30 Mio. Datensätze in der größten Tabelle
- Hardware
  - 144 GB Hauptspeicher
  - 2 * 6 Core CPU 2,4 GHz
- Bis zu 30 Mio. Transaktionen pro Tag (xact_commit)
- Webanwendungen mit bis zu 2.700 aktiven Sessions
- 25 Anwendungsentwicklerinnen und -entwickler

Besonderheiten

- Mixed Workload mit vielen unterschiedlichen Anwendungen (Batch und Dialog)
- Hohe Lastspitzen während der jährlichen Antragsphase
- Häufige und kurzfristige Programmänderungen
- Geodaten und Geoprocessing
- Bitemporale Historisierung
- Zugriff aus Java-Anwendungen über Objektrelationales Mapping (ORM)
- Large Objects
  (allerdings nicht in großem Umfang)
- Betrieb am Rechenzentrum Süd des Freistaats Bayern
- Derzeit PostgreSQL 9.0, PostGIS 1.5

Migration

Praxiserfahrungen einer direkten Anwendungsmigration

- Migration einer größeren Java-Anwendung nach PostgreSQL
  - 2.000 Klassen
  - 400.000 LOC
  - Bisher nur im Test, aber noch nicht produktiv

- Grundsätzliche Kompatibilität von DDL und SQL vorhanden, aber kleine Unterschiede im Verhalten erfordern kleinere Anpassungen

Unterschiede DDL

- Alle bei uns am Großrechner verwendeten Datentypen werden unterstützt, aber:

- Unterschiede bei zulässigen Namen von Indexen
- Unterschiede bei bestimmten Grants

Unterschiede SQL

- Standard-SQL wird weitgehend identisch unterstützt, aber:

- Syntaxunterschiede:
  - nur `CURRENT_TIMESTAMP`, nicht `CURRENT TIMESTAMP`
- Verhalten:
  - Sortierreihenfolge in manchen Fällen anders
  - `GROUP BY` führt in DB2 automatisch zu Sortierung
  - datepart('year', '2100-12-31 24:00:00.000000') != 2100
  - `RTRIM()` verhält sich anders: bei DB2 ist der folgende Vergleich bei Charakterfeldern mit fixer Länge wahr:
    `RTRIM(T1.STATUS) = 'geprüft      '`

Unterschiede JDBC

- Durch JDBC ist grundsätzlich hohe Kompatibilität vorhanden, aber:

- Automatische Konvertierung von Spaltennamen in Kleinbuchstaben führte zu Problemen bei der Nutzung von `ResultSetMetaData`
- `statement.setString()` geht im JDBC-Treiber nicht mehr für beliebige Datentypen
- `SELECT COUNT` liefert bei PostgreSQL `Long` und kann nur mit `getLong()` abgefragt werden (früher `getInt()`)

Unterschiede Daten

- Hex-0-Werte in Characterfeldern führten bei PostgreSQL zu Problemen

Unterstützung bei einer Migration

- Im PostgreSQL-Wiki gibt es „Conversion Guides“ für die Migration von verschiedenen Datenbanksystemen nach PostgreSQL:

  http://wiki.postgresql.org/wiki/Converting_from_other_Databases_to_PostgreSQL

Erfahrungen

Subjektive Einschätzung zur Anwendungsmigration

- Java-Anwendungen mit JDBC und „einfachem“ SQL können mit vertretbarem Aufwand migriert werden, aufgrund
  - Standard-API JDBC
  - SQL-Standardkonformität von PostgreSQL

- Allerdings gab es mehr Probleme als erwartet und ausführliches Testen ist unbedingt nötig

Betriebliche Erfahrungen

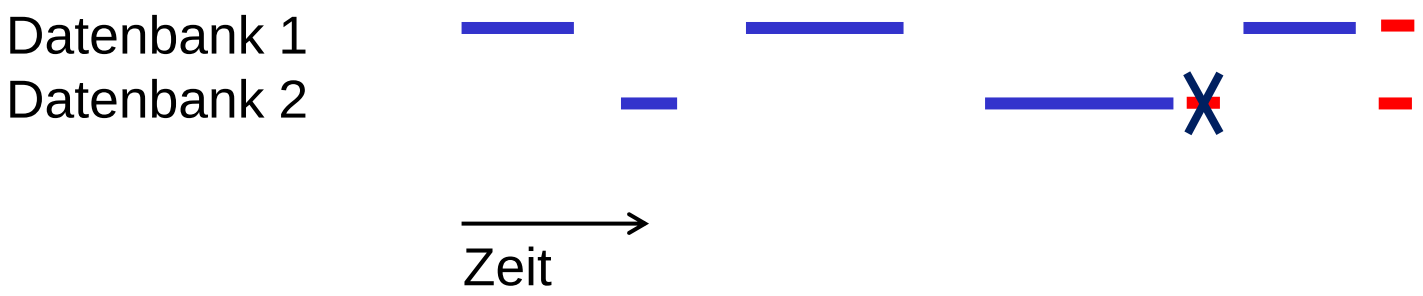
- Ein großer Hauptspeicher steigert die Leistung erheblich
  - bei offenen Systemen sehr preiswert
  - speziell bei lesenden Statements
- PostgreSQL ist aus Sicht der Anwendungsentwicklung problemloser als eine Großrechnerdatenbank
  - erfordert weniger manuellen Administrationsaufwand
  - hat weniger Störungen aufgrund von Wartungsarbeiten

Migration bei einer großen Menge von Anwendungen

- Falls eine komplette Migration nicht möglich ist, dann ist ein praktikabler Weg, während einer Übergangszeit zwei Datenbanken parallel zu pflegen
- In unserem Fall wurden hierzu zuerst die schreibenden Anwendungen migriert, die dann aus der Anwendung heraus beide Datenbanken pflegen
- Bei diesem Vorgehen muss man allerdings mit hoher Wahrscheinlichkeit Programme zur Prüfung bzw. Wiederherstellung der Datenkonsistenz schreiben

Two-Phase-Commit

- Wir haben mit Two-Phase-Commit getestet und es scheint grundsätzlich zu funktionieren, allerdings ist es aus verschiedenen Gründen bei uns bisher nicht in der Produktion aktiv
- Commit-Handling bei zwei Datenbanken ohne Two-Phase-Commit:



1 Millisekunde sind 1.000.000 Nanosekunden

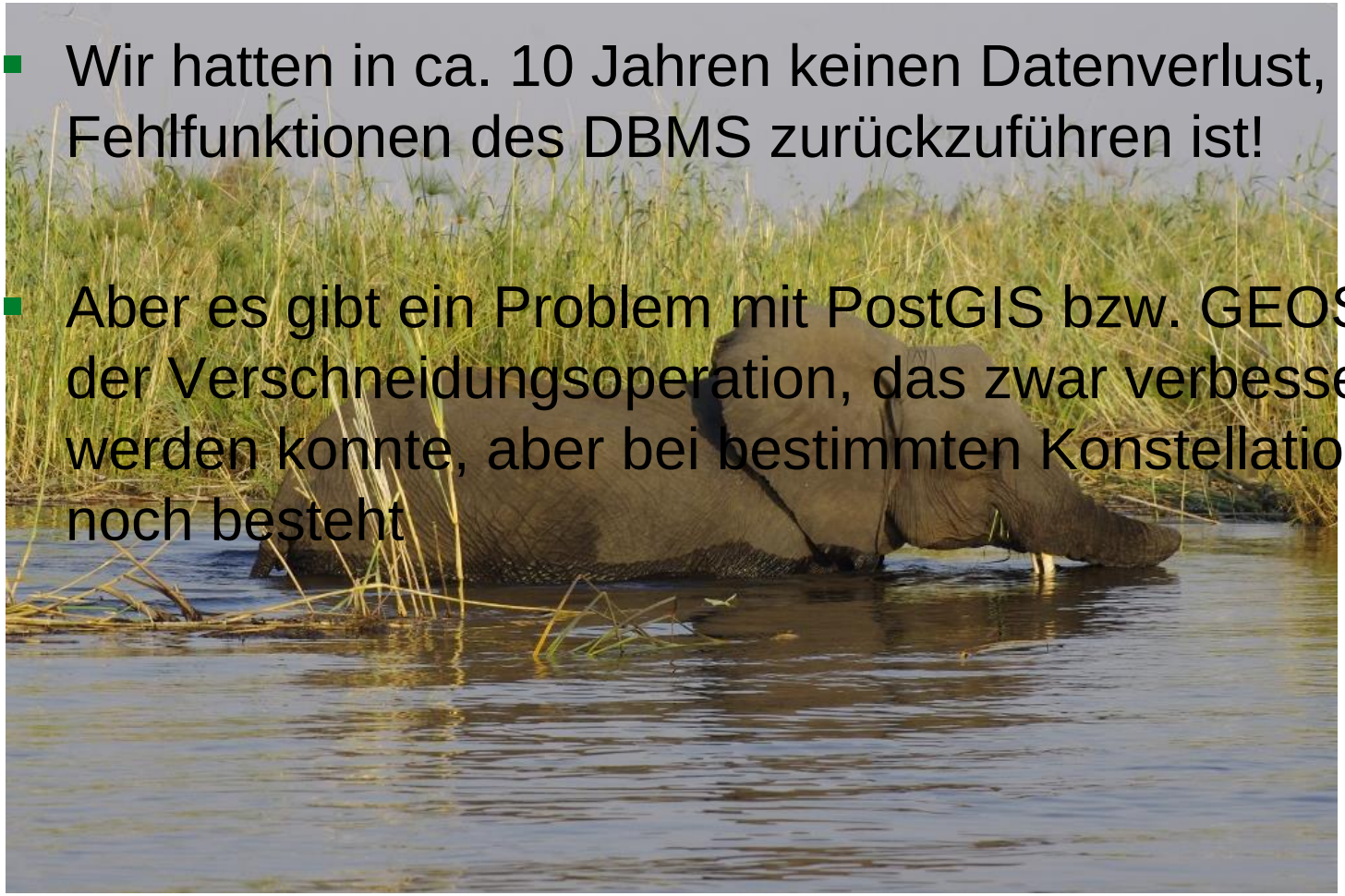
- Oder: Latenzzeiten sind wichtig!
- Latenzzeiten sind die durch die Laufzeit eines Datenpakets im Netzwerk verursachten Verzögerungszeiten (zwischen Client und Server)
- Komplexe Anwendungen können pro Dialogschritt viele Datenbankabfragen haben
- Auf einem Großrechner sind zwischen Anwendung und Datenbank praktisch keine Latenzzeiten vorhanden
- In der offenen Welt müssen Latenzzeiten leider oft in Millisekunden gemessen werden, die sich addieren

Beispielrechnung für Latenzen bei einem Batchlauf

120.000 zu berechnende landwirtschaftliche Betriebe
* 200 SQL-Abfragen pro Betrieb
* 4 IP-Pakete pro Abfrage
* 1 ms Latenz pro IP-Paket

= 96.000.000 ms
= mehr als 26 Stunden zusätzliche Laufzeit durch 1 ms Latenz

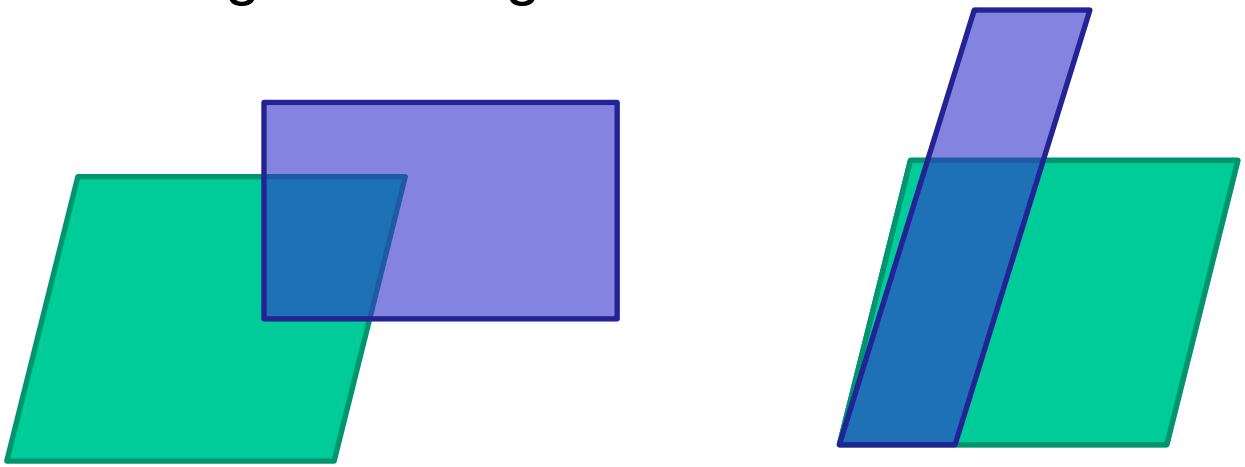
Probleme?



- Wir hatten in ca. 10 Jahren keinen Datenverlust, der auf Fehlfunktionen des DBMS zurückzuführen ist!
- Aber es gibt ein Problem mit PostGIS bzw. GEOS und der Verschneidungsoperation, das zwar verbessert werden konnte, aber bei bestimmten Konstellationen noch besteht

Verschneidungsproblem von Polygonen

- Es gibt für die Verschneidung von Polygonen offenbar keinen numerischen Algorithmus, der bei fast parallelen Kanten gleichzeitig robust und schnell ist



- Nachdem aber bei der Agrarförderung gewisse Toleranzen erlaubt sind, konnten wir das Problem durch die Verwendung von Buffern umgehen.

Bitemporale Historisierung

- Einfache Historisierung:

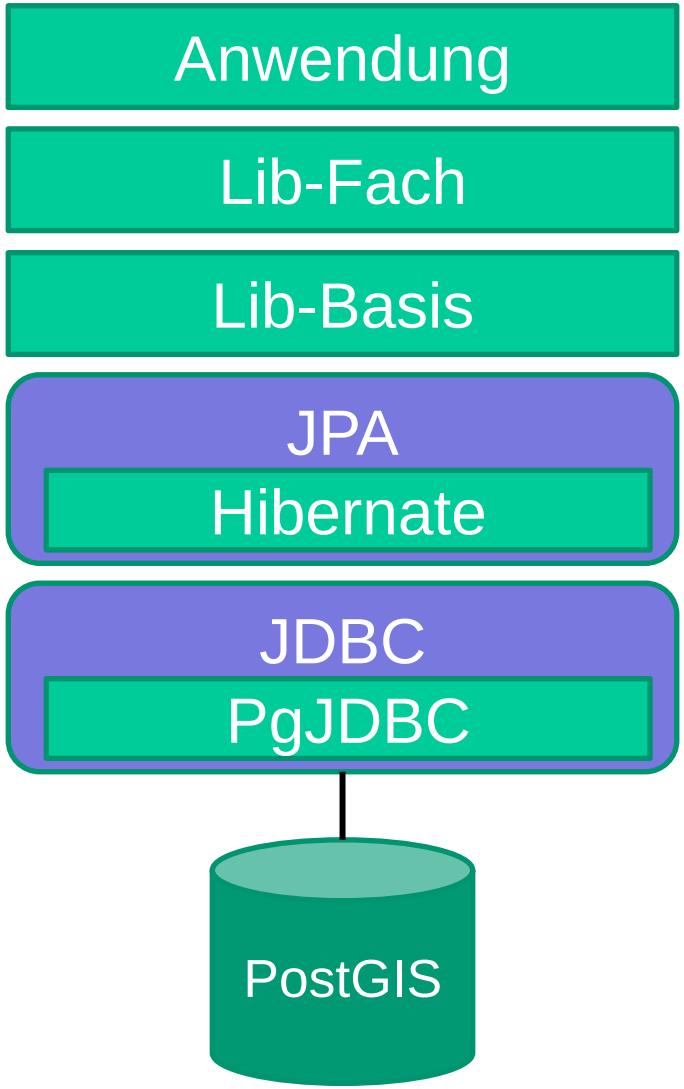
Gültig ab	Gültig bis	Betrieb	Name
2008	2011	1234	Anna Huber
2012		1234	Anna Mayr

- Bitemporale Historisierung:

Gültig ab	Gültig bis	ab	bis	Betrieb	Name
2008	2011	01.01.2008	31.03.2012	1234	Anna Huber
2012		31.03.2012	17.04.2012	1234	Anna Meier
2012		17.04.2012		1234	Anna Mayr

- Richard T. Snodgrass: *Developing Time-Oriented Database Applications in SQL*

Anwendungsarchitektur für Java-Neuentwicklungen



Administrative Attribute in bitemporalen Tabellen

- Automatisch gepflegt durch Framework

Spalte	Datentyp	Bemerkung
ident	int8	eindeutige ID pro Satz
vnr	int2	Versionsnummer für optimistisches Locking
fachvon	date	Beginn der fachlichen Gültigkeit
fachbis	date	Ende der fachlichen Gültigkeit
sysan	timestamp	Beginn der technischen Gültigkeit
sysaus	timestamp	Ende der technischen Gültigkeit
quelle	varchar	Benutzer oder Batchanwendung des Erstellers
abschluss	varchar	Benutzer oder Batchanwendung des Ändernden
refid	Int8	eindeutige ID pro Objekt (für Foreign Keys)

Erfahrungen mit objektrelationalem Mapping

- Sieht einfach aus
- Ist es auch

- Allerdings ist es extrem leicht, Anwendungen sehr ineffizient umzusetzen

- Aber: mit Erfahrung geht es auch effizient!

Tool-Empfehlung

- Wir haben sehr gute Erfahrungen gemacht mit dem Datenbanktool SQL Workbench/J
  - FOSS (Apache 2.0 Lizenz)
  - JDBC-basiert und somit datenbankunabhängig
  - Viele nützliche Features
    - Erzeugung von DDL aus Datenbank
    - Batch mode
  - Besonders geeignet für den gleichzeitigen Umgang mit mehreren Datenbanken
    - Vergleiche von Tabellen
    - Kopieren von Tabellen

Feature-Wunsch aus der Praxis

- Das Problem:
  - Viele Queries auf mittelgroße Tabellen, bei denen die benötigten Indexe nicht vorhanden sind und die deshalb einen Full Table Scan machen
- Eine mögliche Lösung:
  - Aufspüren solcher Queries durch PostgreSQL:
    - Bei der Planerstellung kein Index vorhanden
    - Table Scan geht über größere Tabelle
    - ➔ Verdächtiges Statement protokollieren
  - Wegen Laufzeitverlängerung eventuell nur in einem temporären Analysemodus?

Ausblick

- PostgreSQL wird für uns noch wichtiger werden:
  - Cluster mit standortübergreifender Replikation
  - Ablösung des Großrechners und Migration von Natural nach Java und DB2 nach PostgreSQL
  - Speicherung von Dokumenten in der Datenbank, die aktuell noch im Dateisystem abgelegt werden

Vielen Dank für Ihre Aufmerksamkeit!

!?