BigData-Integration in eine BI-Landschaft per Data Vault

Frank Stoll, Philipp Hildebrandt
GfK SE
Nürnberg

Schlüsselworte
Datenintegration, Data Vault

Disclaimer

Der Vortrag beschäftigt sich mit der Integration von Handelsdaten eines Hadoop-Systems mit Stammdaten eines bestehenden Systems mithilfe der Data Vault-Modellierungsmethodik. Das Laden von Daten aus dem Hadoop-System ist nicht Gegenstand dieses Vortrages. Beide Systeme sind durch eine Message-Queue und einen Fileshare entkoppelt. Es erfolgt kein direkter Zugriff auf das Hadoop-System.

Einleitung
Mit einem Umsatz von 1543,4 Millionen Euro, mehr als 13.000 Mitarbeitern und der Präsenz in mehr als 100 Ländern ist die GfK eines der größten Marktforschungsunternehmen weltweit. Dabei werden zwei Sektoren, „Consumer Experiences“ (warum Konsumenten etwas kaufen) und „Consumer Choices“ (was Konsumenten tatsächlich wann und wo gekauft haben) unterschieden.

Um letzteres Verbraucherverhalten zu messen, müssen die Verkaufszahlen erfasst, normiert, qualitätsgeprüft und am Ende für Produkte - in Form von interaktiven Reports oder auch Dashboards - optimiert aufbereitet werden. Um den wachsenden Datenmengen zu begegnen, wird basierend auf Hadoop eine neue Datenproduktionsstrecke entwickelt. Deren Ziel ist es, hochautomatisiert und geschwindigkeitsoptimiert die Verkaufszahlen für die GfK-Kundenprodukte aufzubereiten und zur Verfügung zu stellen.
Die Anbindung der neuen Hadoop-basierten Datenproduktionsstrecke an das bestehende GfK-Datenproduktions- und Reporting-System erfordert es, Schnittstellen zur Integration neuer und bereits vorhandener Systembestandteile zu definieren. Im hier beschriebenen Fall wurde diese Schnittstelle zur Integration von Daten aus verschiedenen Systemen in Form eines Data Warehouses mit der Modellierungsmethode Data Vault auf Basis von Oracle implementiert.

Aufgabe

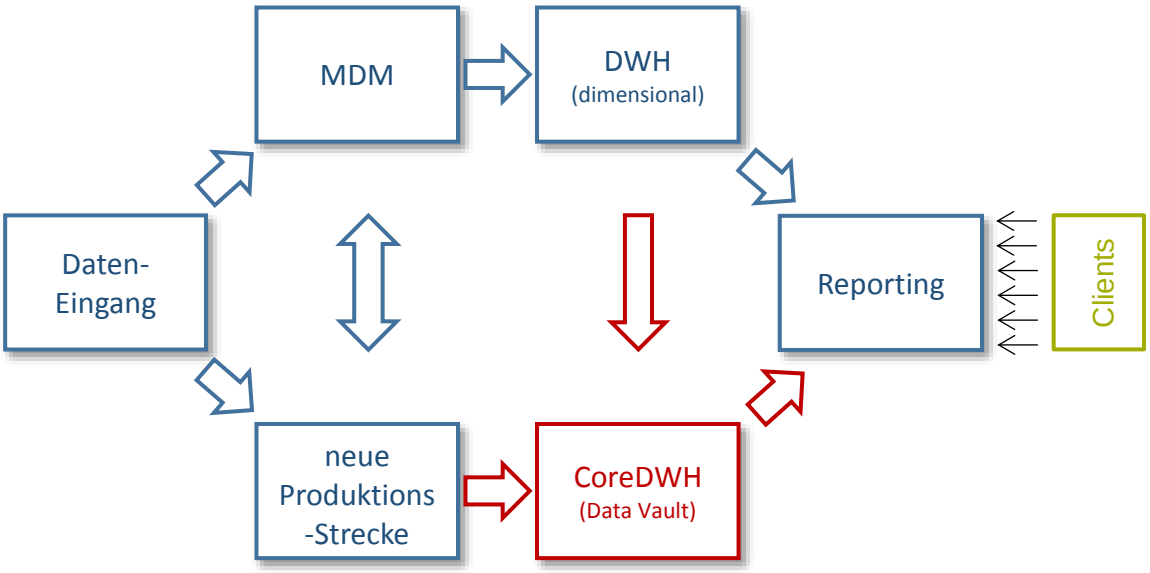

Abbildung 1: Integration eines „Schnittstellensystems“ in bestehende Landschaft

Die o.g. neue Produktionsstrecke soll in die bestehende Systemlandschaft integriert werden, so dass diese möglichst wenig beeinflusst und trotzdem ein zeitnaher erster Durchstich ermöglicht wird.

Als Schnittstelle zwischen bestehender und neuer Welt wurde das in Abbildung 1 rot dargestellte CoreDWH implementiert. Im Gegensatz zum bestehenden dimensional modellierten DWH wurde es mit der Data Vault Methode modelliert und soll eine schrittweise Integration der neuen Produktionsstrecke in die bestehende Systemlandschaft ermöglichen. Wie in der Abbildung erkennbar, existiert (noch) keine direkte Schnittstelle vom MDM System zum CoreDWH. Der Grund hierfür ist, dass in der Schnittstelle zum bestehenden DWH Geschäftsregeln implementiert sind, die zum jetzigen Zeitpunkt nicht in einer neuen Schnittstelle implementiert werden sollen.

Data Vault

Die durch Dan Lindstedt in den 1990ern entwickelte und 2000 offiziell vorgestellte Datenmodellierungsmethode Data Vault wurde für die Modellierung eines Data Warehouses entwickelt. Sie besteht im Wesentlichen aus drei Komponenten, genannt Hub, Satellit und Link. Ein Hub enthält den unveränderlichen identifizierenden Schlüssel einer Entität, der Satellit die veränderlichen beschreibenden Nicht-Schlüsselattribute und der Link die Beziehung zwischen zwei oder mehr Hubs. Auch Links besitzen Satelliten, welche beispielsweise die Gültigkeit einer Beziehung beschreiben.

Entwicklung des Data Vault Modells im Projekt

Data Vault hat neben seinem hohen Anspruch an Flexibilität starke Vorgaben in Richtung Standardisierung. Die Standardisierung schreibt für jede Komponente - Hub, Link und Satellit - eine Menge an Meta-Attributen vor. Diese bestehen in der Mindestausprägung aus einem Surrogat-Schlüssel (ausgenommen der Satellit), aus dem Ladezeitpunkt und Angaben über die Quelle eines geladenen Datensatzes. Neben diesen vorgeschriebenen Metadaten wurden im Projekt weitere Meta-Attribute und eine strenge Namenskonvention für diese definiert. Da diese Standardisierung und Konvention für jede Komponente gilt, wurde sie zu Beginn der Modellierung festgelegt, um den späteren Änderungsaufwand zu minimieren. Diese Konventionen führen dazu, dass die Komponenten für den Ladeprozess weitgehend automatisiert erstellt werden können. Siehe dazu weiter unten in Kapitel Ladeprozess. Im Projekt wurde auf ein Ende-Attribut in den Satelliten verzichtet, um auf Update-Operationen vollständig verzichten zu können. Dieser Ansatz bietet zum einen den Vorteil der besseren Ladegeschwindigkeit, zum anderen ist der Ladeprozess damit auch relativ einfach und unkompliziert. Gelöschte Datensätze werden als “leerer“ Datensatz geladen und mit einer selbstdefinierten Standard-Spalte, einem Löschkennzeichen, markiert.

Neben der Möglichkeit der hohen Automatisierung stellt Data Vault eine sehr flexible Möglichkeit der Modellierung dar, welche die in diesem Projekt genutzte agile Vorgehensweise sehr gut unterstützt. Durch die Entkoppelung von Hubs durch Links kann ein Modell schrittweise entwickelt werden, ohne in jeder Iteration des Entwicklungsprozesses bereits implementierte Beziehungen aufbrechen oder fortwährend Daten migrieren zu müssen.

Architektur

Neben der eigentlichen Modellierung wurde für den Zugriff auf Daten die in den neuen Systemen entwickelte Standard-Zugriffsarchitektur als Vorlage gewählt und entsprechend angepasst.

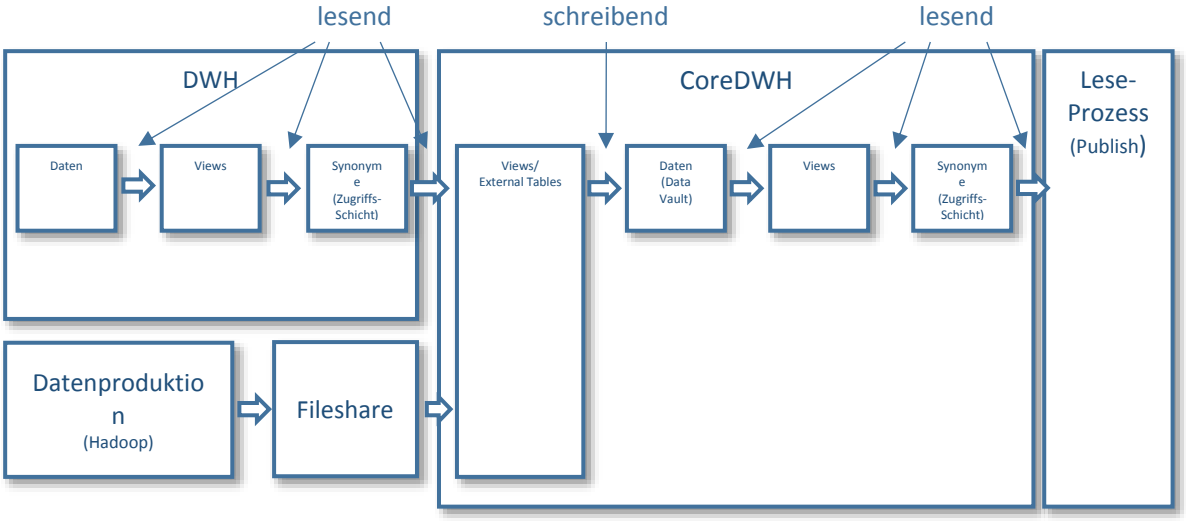

Abbildung 2: Zugriffs-Architektur

Wie in Abbildung 2 zu erkennen, erfolgt ein Zugriff auf Daten der jeweiligen Quellsysteme nicht direkt. Stattdessen wird der lesende Zugriff durch eine Zugriffsschicht (Schema) ermöglicht, in welcher sich lediglich Synonyme auf die darunter befindliche View-Schicht befinden.

In der aktuellen Phase erfolgt keine direkte Integration des Hadoop-Systems. Dieses stellt die produzierten Daten in einem File-Share als csv-Datei zur Verfügung, auf welche wiederum durch eine *Oracle External Table* zugegriffen wird.

Zugriff auf das Data Vault Modell

Die im Data Vault Modell gespeicherten Daten werden durch den im bestehenden System eingesetzten Publish-Prozess ausgelesen und an das Reporting-System weitergegeben. Eine der Rahmenbedingungen des Projektes ist, dass der bestehende Publish-Prozess unverändert eingesetzt werden kann. Hierfür muss diesem eine Datenstruktur zur Verfügung gestellt werden, welche sich nicht von der alten Struktur unterscheidet. Um dies zu ermöglichen, wurde mehrstufig verfahren. Die erste Stufe beinhaltet eine View-Schicht im Data Vault, welche Hub bzw. Link und zugehörige Satelliten zusammenfasst und in jeweils einer View als vereinfachte Schicht für lesende Zugriffe bereitstellt. Da in Data Vault die Historie sämtlicher Veränderungen der Entitäten gespeichert wird und durch das Paradigma „INSERT ONLY“ kein Datensatz explizit durch ein Datum beendet werden kann, wird in diesen Views das Enddatum zur Laufzeit berechnet. Hierfür wird die analytische Windowing-Funktion *lead* verwendet.

```
LEAD (load_date) OVER (PARTITION BY hub_id ORDER BY load_date) AS
valid to
```

Listing: 1: lead-Funktion zur Berechnung des Endedatums zur Laufzeit

Anschließend werden in der in Abbildung 2 dargestellten View-Schicht des Core-DWH die Daten so strukturiert, dass der Publish unverändert die gleichen Strukturen wie im bestehenden DWH vorfindet.

Der Zugriff auf diese Strukturen erfolgt durch den Publish-Prozess über die Zugriffsschicht des CoreDWH.

Ladeprozess

Die Grundidee des Ladeprozesses ist, diesen möglichst transparent und nachvollziehbar zu gestalten. Hierzu wird der Datentransport von der Transformationslogik getrennt. Zur Reduktion der Komplexität und zur verbesserten Wartbarkeit wird ein Großteil der Ladeobjekte automatisch generiert. (Basierend auf Vollabzug der Daten!)

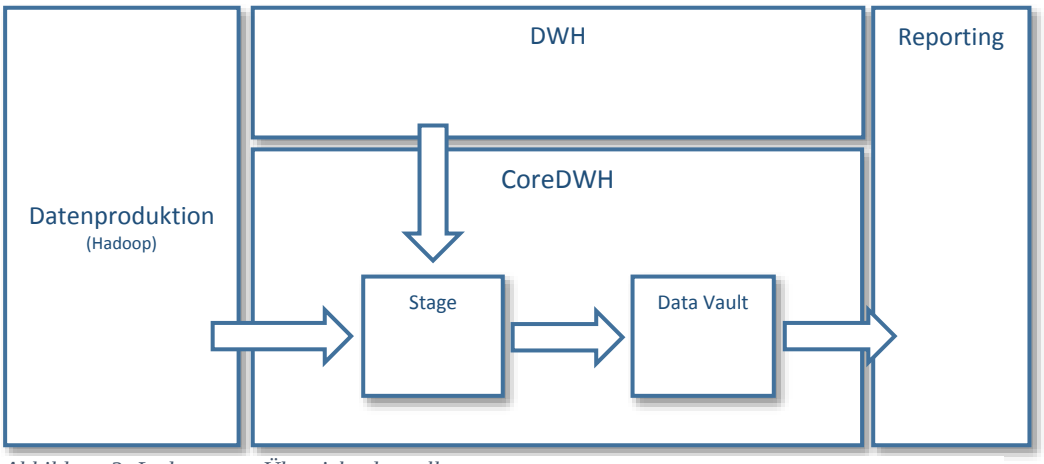

Abbildung 3: Ladeprozess Übersichtsdarstellung

Extraktion

Die Extraktion der Daten erfolgt je nach Vorsystem entweder über eine *Oracle External Table* vom Fileshare oder über einen *Database Link* in das bestehende DWH System. Diese werden in einer View-Schicht, den sogenannten External Views, gekapselt. Im Rahmen der Entwicklung wurden in der View-Schicht Konsistenzprobleme des Vorsystems durch teilweise umfangreiche Joins zur Filterung von Daten aufgelöst, Datenaufbereitungen durchgeführt und die Data Vault Meta-Attribute hinzugefügt.

Diese Methode widerspricht dem Paradigma von Data Vault, die Daten unverändert zu Laden. Dies wurde im Rahmen der Entwicklung in Kauf genommen und wird derzeit wieder umgebaut, was teilweise deutlich Performanceverbesserungen ermöglicht.

Transformation

Die Transformationslogik wird, wie im vorherigen Kapitel bereits angeschnitten, in der Datenbank auf Basis von Views abgebildet. Dies ermöglicht es, direkt in der Datenbank alle Transformationsprozesse analysieren zu können.

Im gesamten Ladeprozess gibt es drei View-Schichten. Die erste Schicht ist die External View Schicht. Diese bindet die externen Objekte ein, harmonisiert die Attribut- und Objektnamen und bereitet so den Stage-Load vor.

Die darauf folgende Schicht dient als Quelle zum Beladen des Data Vault. Sie transformiert die Daten der Stage Schicht derart, dass durch das Übertragen aller Daten der View der neue Datenstand

abgebildet wird. Dies bedeutet bei Hubs und Links ein Einfügen aller neuen Daten, bei Satelliten das Einfügen von neuen und geänderten Datensätzen und das Erstellen von Löschsätzen für nicht gelieferte Daten.

Die letzte Schicht berechnet, wie oben bereits beschrieben, für die Data Vault Objekte den jeweils gültigen Datenstand.

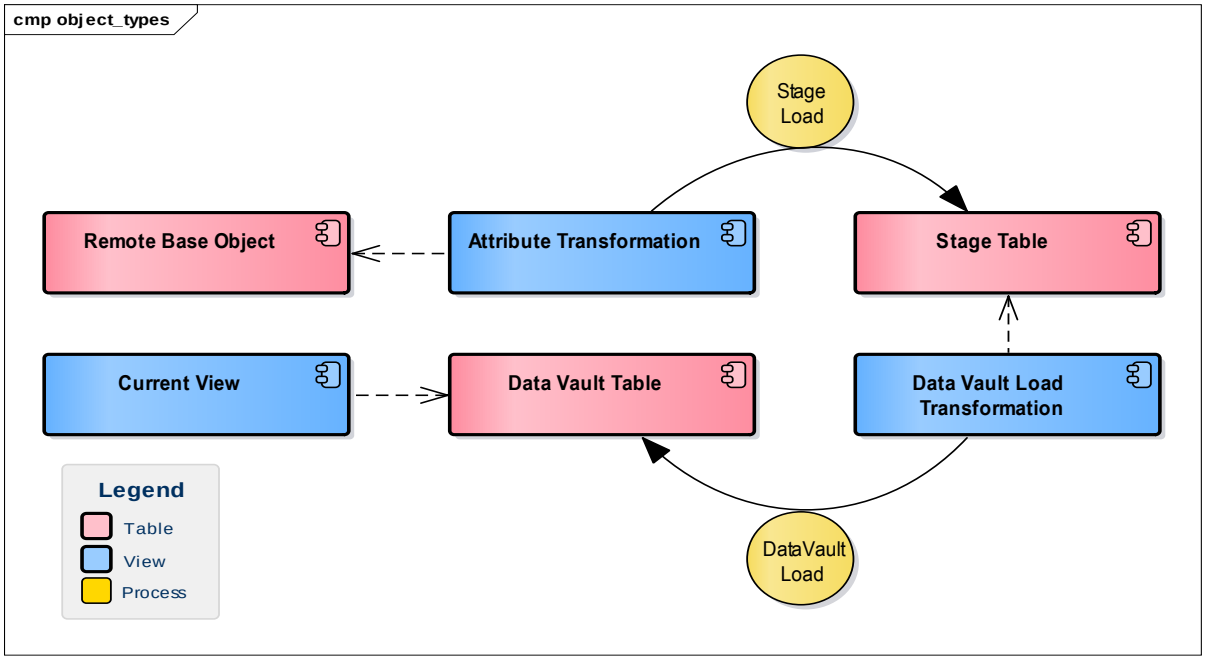

Abbildung 4: Objekttypen in der Ladestrecke

Automatisierung

Eine Eigenheit von Data Vault ist es, das Logische Datenmodell in sehr viele Tabellen zu zerlegen. Die von uns zusätzlich eingeführten Datentransformationsschichten vervielfältigen die Anzahl der Datenbankobjekte. Um die Komplexität und den Aufwand beherrschbar zu halten, sollten möglichst viele der Objekte des Ladeprozesses automatisch erzeugt werden. Als Basis hierfür dienen die manuell erstellten Data Vault Tabellen und die External Views.

Die Staging Tabellen können einfach per `CREATE TABLE AS SELECT` aus den External Views generiert werden. Die Objekte der letzten Viewschicht lassen sich ebenfalls unkompliziert aus dem Satelliten und dessen Hub oder Link und dem oben genannten `lead`-Ausdruck generieren.

Die Generatoren für die Data Vault Ladeviews nutzen die im Projekt festgelegten Namens-konventionen und das Data Dictionary der Datenbank. Diese werden verwendet, um zusammengehörige Attribute zu verbinden und die Tabellenabhängigkeiten aufzulösen.

Zusätzlich wurde ein Metadatenmodell eingeführt, welches die einzelnen Datenbankobjekte zu Teilladestrecken verknüpft und eine Generierung der gesamten Ladeobjekte und Ladejobs erlaubt.

Beladung und Jobsteuerung

Durch die Kapselung der Transformationslogik in der Datenbank kann das Ladetool von dieser befreit und vereinfacht werden. Die Datentransformation wird als Bestandteil des Transfers aller Daten der jeweiligen Quellview in die passende Zieltabelle realisiert. Beim Beladen der Staging-Tabelle wird diese zuvor mittels *truncate* geleert, um nur neue Daten zu laden.

Die Data Vault Beladung erfolgt entsprechend der Abhängigkeiten zwischen den Tabellentypen in mehreren Stufen. Zuerst werden die Hubs, anschließend die Links und zuletzt die Satelliten beladen. Im Anschluss werden die Business Regeln verarbeitet und der Nachfolgeprozess, der Publish, angestoßen. Innerhalb der einzelnen Stufen können alle Ladeschritte parallel ausgeführt werden, da diese unabhängig voneinander sind.

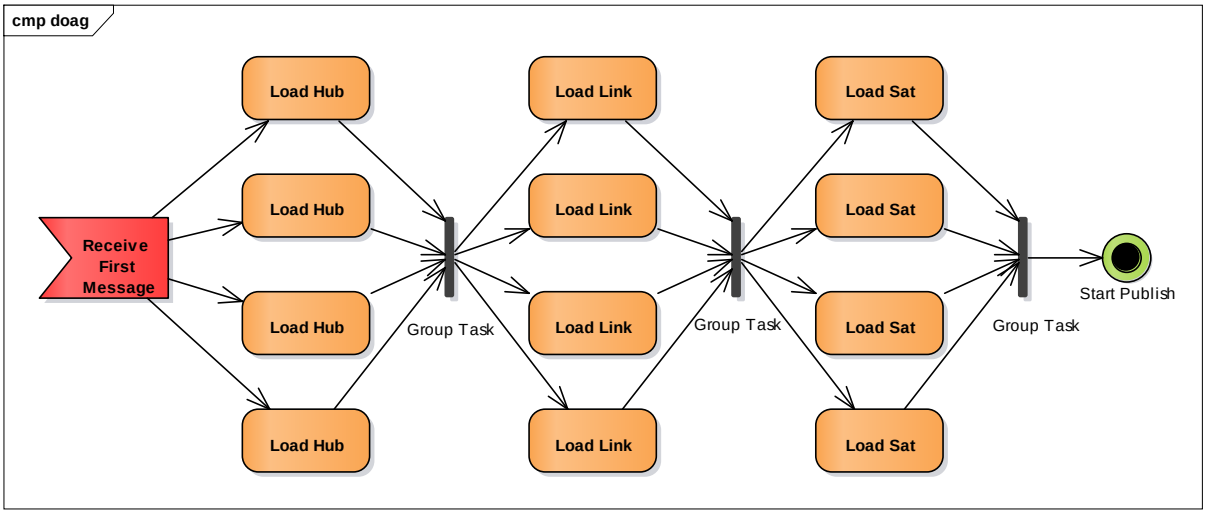

Abbildung 5: Stufen der Data Vault Ladestrecke

Die Parallelisierung der verschiedenen Ladeschritte wird dabei vom Ladetool, welches in Java implementiert ist, realisiert. Dieses liest mit parallelen Workern alle aktuellen Tasks aus einer JMS-Queue aus und arbeitet diese ab. Um die Ladestufen in der Queue nutzen zu können, werden die einzelnen Stufen durch Synchronisierungstasks getrennt. Beim Abarbeiten dieser Tasks wird geprüft, ob alle Tasks der aktuellen Stufe abgeschlossen sind. Anschließend werden die Tasks der nächsten Stufe aus den Metadaten gelesen und als Nachrichten eingestellt. Somit sind immer nur die Tasks der aktuellen Stufe zur Verarbeitung verfügbar.

Als JMS-Queue wurde die *Oracle Advanced Queue* der Datenbank verwendet, was zusammen mit dem Laden des Java-Tools in die Datenbank einen in sich abgeschlossenen Betrieb und bessere Interaktion mit weiteren PL/SQL Prozessen ermöglicht.

Fazit

Die Erfüllung der Anforderung, innerhalb kürzester Zeit ein neues Hadoop-System in eine bestehende komplexe Systemlandschaft zu integrieren und erst während des laufenden Entwicklungs-Prozesses die genauen fachlichen Anforderungen zu erarbeiten, wurde durch die strenge Entkopplung der unterschiedlichen Systeme und Technologien, die hohe Flexibilität der Data Vault Modellierungsmethodik und des hohen Automatisierungsgrades des Ladeprozesses ermöglicht.

Kontaktadressen:
Philipp Hildebrandt
GfK SE
Nordwestring 101
D-90419 Nürnberg

Telefon: +49 (0) 911-395 4516
E-Mail philipp.hildebrandt@gfk.com
Internet: www.gfk.com

Frank Stoll
GfK SE
Nordwestring 101
D-90419 Nürnberg

Telefon: +49 (0) 911-395 2779
E-Mail frank.stoll@gfk.com
Internet: www.gfk.com