noETL yesSQL

Warum ELT und SQL die optimale Wahl für ein modernes DWH ist!

Alec Shalashou

28.06.2018

data sqill

Buzzwords Explained

noETL = Ausführung von Transformationen dort, wo die Daten gespeichert sind

yesSQL = Entwicklung der Transformationslogik mit SQL

© 2018 SoftQuadrat GmbH

Inhalt

- Aktuelle DWH Trends
- noETL
  - ETL früher und heute
  - ELT Ansatz
  - Datenbank als ELT Platform
- yesSQL
  - SQL als Transformationssprache
  - SQL vs Point-and-Click
- noETL yesSQL in modernen DWHs
  - Transformationslandschaft
  - Best Practices

© 2018 SoftQuadrat GmbH

Aktuelle DWH Trends

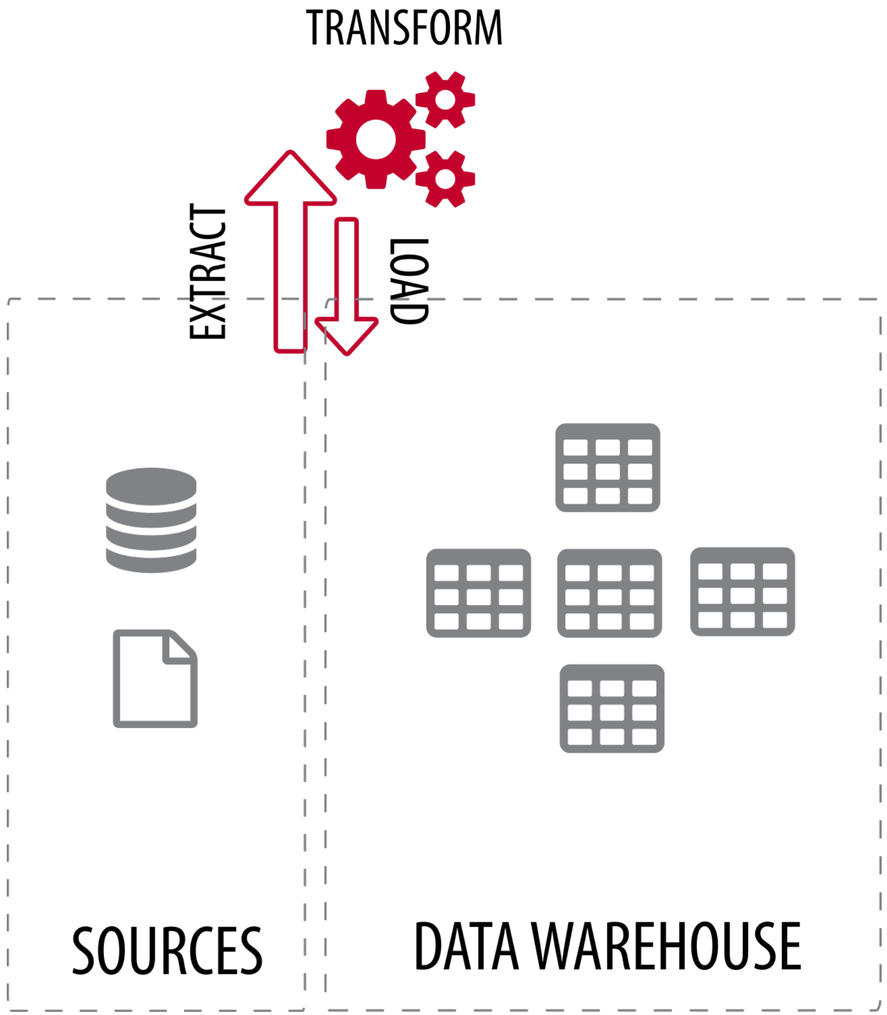
- Datenmengen wachsen
- Entwicklung wird agiler
  - Enwicklungszyklen werden kürzer
  - Prototyping
- Lebensdauer von Ergebnissen (Reports, Analysen) wird kürzer
- Komplexität der analytischen Fragestellungen wächst
- Auch SMBs haben hohe analytische Anforderungen
- Bedeutung von Cloud-Lösungen wächst weiter an

© 2018 SoftQuadrat GmbH

noETL

© 2018 SoftQuadrat GmbH

Erster DWH Ansatz



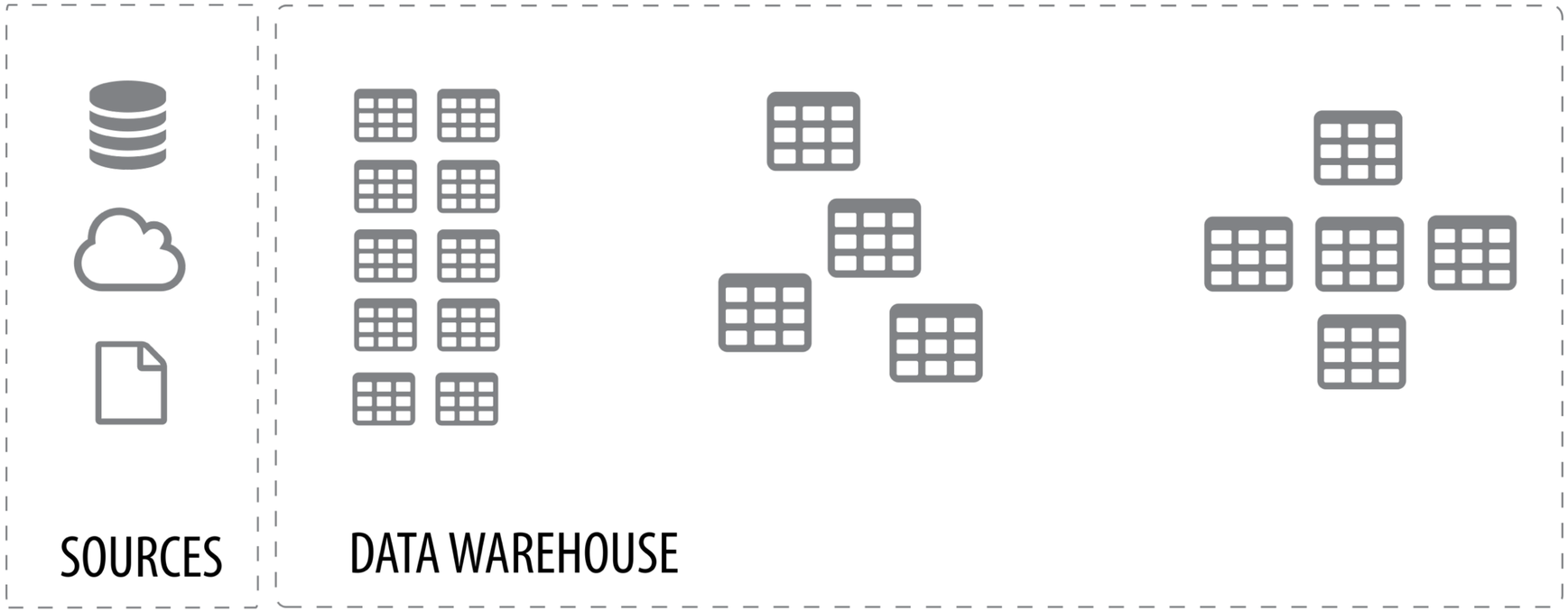
© 2018 SoftQuadrat GmbH

Warum hat man ETL gemacht?

- Direkter Weg - ein Integrationsserver wurde benötigt um Daten zwischen Systemen zu schieben
- Datenbankspeicher war begrenzt und teuer, deshalb hat man die Daten gleich transformiert
- Der Integrationsserver wurde auch für Transformation mitbenutzt, da die Datenbanken für Massendatenverarbeitung nicht performant waren
- Transformationen/Verarbeitungslogik noch relativ einfach

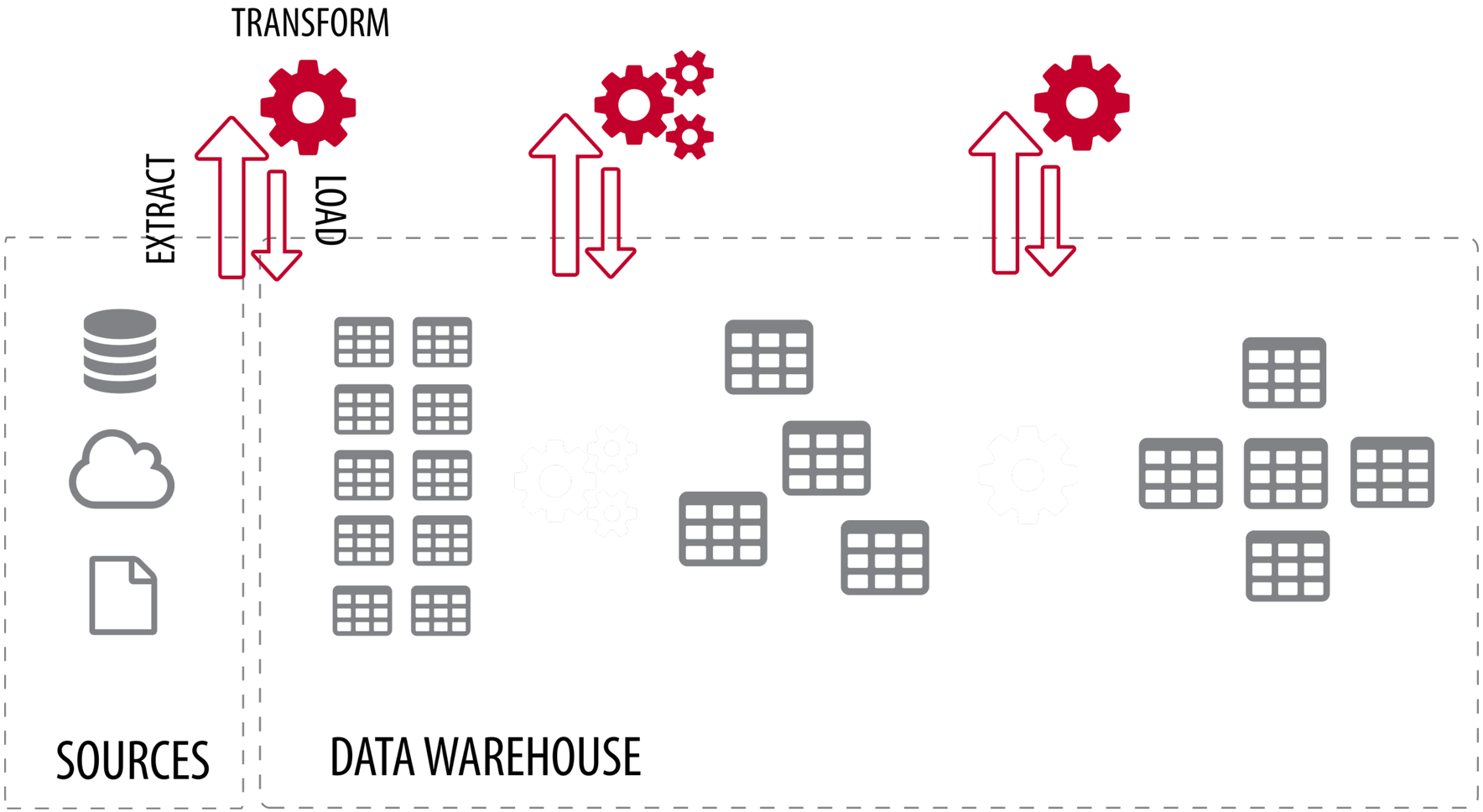
© 2018 SoftQuadrat GmbH

DWH heute



© 2018 SoftQuadrat GmbH

ETL heute



© 2018 SoftQuadrat GmbH

Merkmale von ETL Tools

- Prozess
  - Lesen von Daten aus unterschiedlichen Quellen
  - Transformation auf dem Application Server
  - Schreiben der transformierten Daten in das Zielformat
- Transformationslogik
  - Grafische Modellierung der Transformationslogik, oft mit propriäteren Sprachen
  - manchmal Code Generierung
- Grafische Modellierung der Ablaufsteuerung, manchmal vermischt mit Transformationslogik
- Oft eingebaut Scheduling / Monitoring Komponente

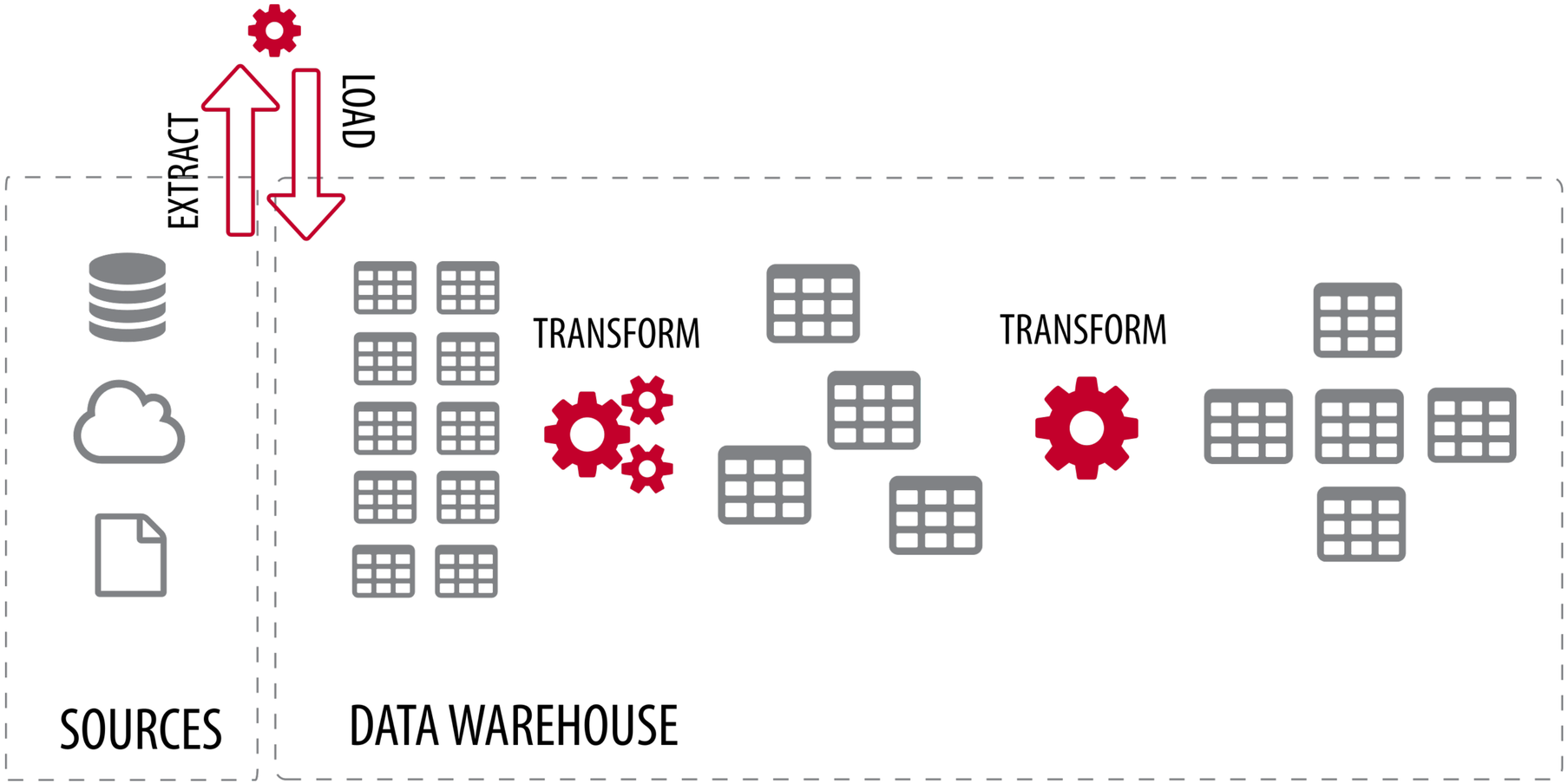
© 2018 SoftQuadrat GmbH

Warum ändern?

- Nicht-optimales Performance-Verhalten
  - Ständiges Daten Rauslesen und Schreiben, wiederholtes Lesen derselben Daten
  - Row-based Verarbeitung ist nicht immer performant für Massendaten
- Oft ungünstig für neue Technologien (Cloud, In-Memory, MPP, Big Data)
- Vendor Lock-In
- Komplexe und intransparente Architektur
- Konkurrenz um Ressourcen zwischen Datenbank und ETL Server, oft ungleichmäßige Auslastung von Ressourcen von ETL Server (Tag vs Nacht)
- Schwergewichtiger Entwicklungsprozess
  - Mitarbeiter müssen grafische Sprache/Code Sprache des ETL Tools kennen
  - Oft Medienbruch zwischen Prototyp und produktiv-laufendem Code (ETL Code) sowie späterer Fehleranalyse

© 2018 SoftQuadrat GmbH

ELT



© 2018 SoftQuadrat GmbH

Warum ELT?

- Performance Verhalten
  - Ist für Massendaten ausgelegt, da die Datenverarbeitung dort stattfindet, wo die Daten gespeichert sind (kein Netztraffik)
  - Ausnutzung der Features der Datenspeicher - essentiell für In-Memory, Cloud, MPP, Big Data
- Transparente Architektur
- Vendor Lock-In nicht so stark, da der Transformationscode portabel bleibt
- Einfachere Entwicklung und Debugging, da der Quellcode der Transformationen direkt in der Zielumgebung ausführbar ist
- Für Entwickler bekannte Transformationssprache (SQL oder Java/Scala) > siehe yesSQL

© 2018 SoftQuadrat GmbH

Datenbanken stellen gute ELT Plattformen dar

- Datenbanken sind gut in Massendatenverarbeitung:
  - optimierte Ausführungspläne
  - MPP
  - Column-based
  - In-Memory
- Speicherplatz ist kein Problem mehr
- Datenbanken bieten Integrationskomponenten:
  - Bulk Loaders
  - CSV, XML, JSON-Parser
  - Streaming
- Datenbanken können in der Cloud betrieben werden und skalieren dort nach Bedarf
- Die Entwicklung ist noch lange nicht abgeschlossen!

© 2018 SoftQuadrat GmbH

© 2018 SoftQuadrat GmbH

Wie sieht es heute mit SQL aus?

- ANSI SQL ist als Standard etabliert und sehr verbreitet
- SQL bietet mächtige Transformationsmöglichkeiten (z.B. analytische Funktionen)
- Neue Cloud SQL-Datenbanken kommen auf den Markt (Amazon Redshift, Snowflake)
- Viele Big Data SQL Tools (Hive, Impala, Spark SQL)
- NewSQL kommt zusätzlich zu NoSQL (Volt DB, Cocroach DB)
- SQL-basierte Flat-File Integrationstools (Apache Drill)
- Die Mehrzahl der BI Tools unterstützen die Verwendung von SQL

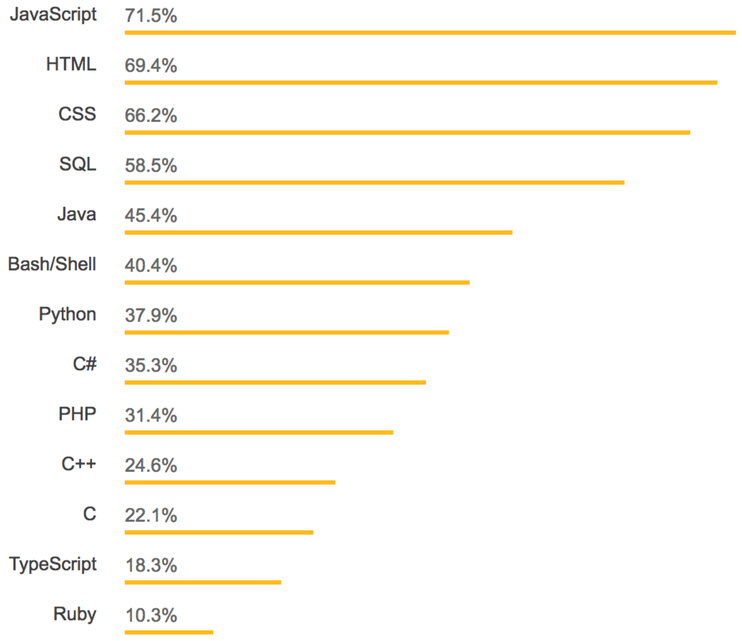
© 2018 SoftQuadrat GmbH

yesSQL - Stackoverflow

Most Popular Technologies

Programming, Scripting, and Markup Languages

All Respondents | Professional Developers



© 2018 SoftQuadrat GmbH

Quelle: https://insights.stackoverflow.com/survey/2018/

Warum SQL für Transformationen?

- Universelle Schnittstellensprache
- Native Transformationssprache für Datenbanken
- Schnelle und agile Entwicklung, da die Analyse, der Prototyp und die Implementierung in der gleichen Sprache erfolgen
- Einfachere Fehleranalyse, da der Transformations-Code direkt ausführbar ist
- Mengensprache

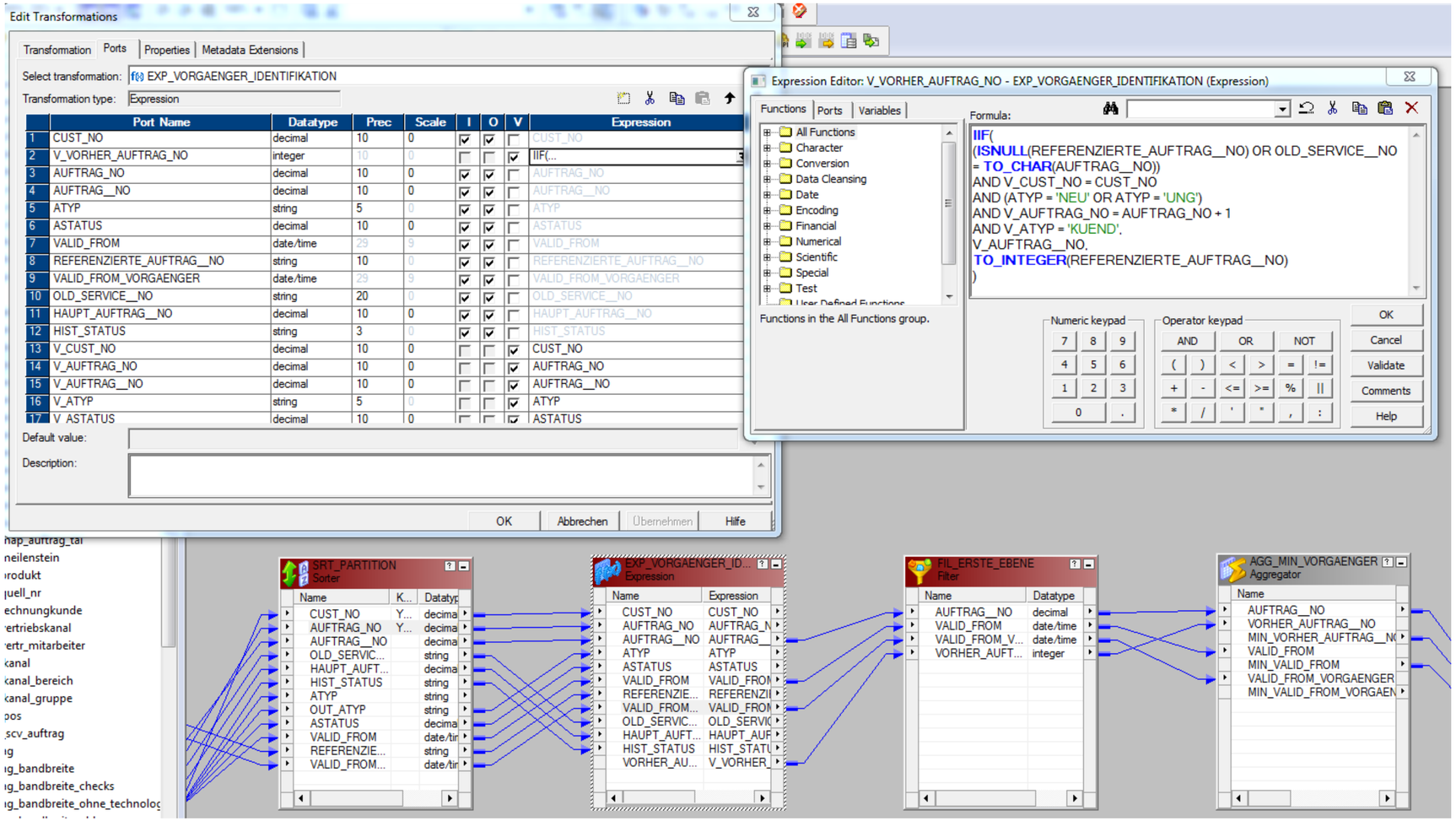
© 2018 SoftQuadrat GmbH

yesSQL vs Point-and-Click Transformationsentwicklung

- Weniger Abhängigkeit vom eingesetzten Tool, da der Inhalt der Transformationen durch SQL portabel ist
- Größeres Angebot an Mitarbeitern mit SQL Kenntnissen
- Schnellere Einarbeitung, da weniger Spezial-Kenntnisse benötigt wird
- SQL ist (in der Regel) kompakter und zugleich übersichtlicher als Point-and-Click Mapping und deshalb besser für komplexere Logik geeignet

© 2018 SoftQuadrat GmbH

yesSQL vs Point-and-Click - Beispiel 1



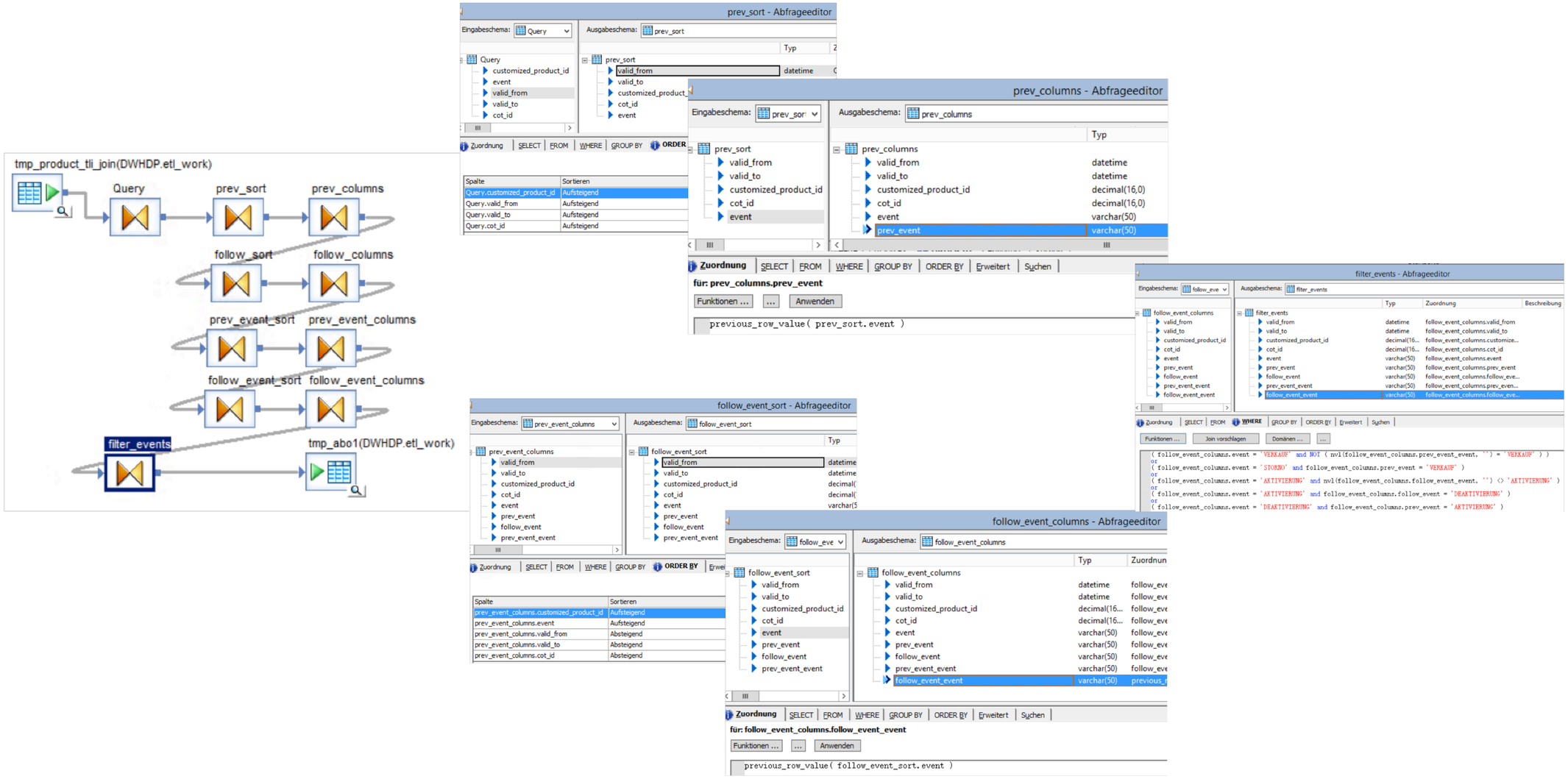
© 2018 SoftQuadrat GmbH

yesSQL vs Point-and-Click - Beispiel 1


```
SELECT auftrag__no
     , MIN(ref_auftrag__no) vorher_auftrag__no
     , MIN(valid_from) gueltig_von
  FROM (SELECT auf.auftrag_no
             , auf.auftrag__no
             , auf.valid_from
             , auf.hist_status
             , TRIM(auf.atyp) atyp
             , auf.astatus
             , auf.cust_no
             , auf.old_service__no
             , ref_auf.min_valid_from
             , ref_auf.max_valid_to
             , ref_auf.ref_atyp
             , CASE
                 WHEN NVL(ref_auf.ref_auftrag__no, auf.auftrag__no) = auf.auftrag__no
                  AND TRIM(auf.atyp) IN ('NEU', 'UNG')
                  AND LEAD(TRIM(auf.atyp)) OVER (PARTITION BY auf.cust_no ORDER BY auf.auftrag_no) = 'KUEND'
                  AND LEAD(auf.auftrag_no) OVER (PARTITION BY auf.cust_no ORDER BY auf.auftrag_no) = auf.auftrag_no + 1
                 THEN LEAD(auf.auftrag__no) OVER (PARTITION BY auf.cust_no ORDER BY auf.auftrag_no)
                 ELSE NVL(ref_auf.ref_auftrag__no, auf.auftrag__no)
               END ref_auftrag__no
          FROM vv_auftrag auf
             , tmp_ref_auftrag ref_auf
         WHERE auf.old_service__no = TO_CHAR(ref_auf.ref_auftrag__no(+)))
 WHERE ref_auftrag__no != auftrag__no
 GROUP BY auftrag__no;
```

© 2018 SoftQuadrat GmbH

yesSQL vs Point-and-Click - Beispiel 2



© 2018 SoftQuadrat GmbH

yesSQL vs Point-and-Click - Beispiel 2


```
SELECT customized_product_id
     , valid_from
     , valid_to
     , cot_id
     , event
  FROM (SELECT customized_product_id
             , valid_from
             , valid_to
             , cot_id
             , event
             , LAG(event, 1) OVER (PARTITION BY customized_product_id ORDER BY valid_from, valid_to, cot_id) prev_event
             , LEAD event, 1) OVER (PARTITION BY customized_product_id ORDER BY valid_from, valid_to, cot_id) follow_event
             , LAG(event, 1) OVER (PARTITION BY customized_product_id ORDER BY event, valid_from, valid_to, cot_id) prev_event_event
             , LEAD(event, 1) OVER (PARTITION BY customized_product_id ORDER BY event, valid_from, valid_to, cot_id) follow_event_event
          FROM etl_work.tmp_product_tli_join) a
 WHERE (event = 'VERKAUF' AND NOT (NVL (prev_event_event, '') = 'VERKAUF'))
    OR (event = 'STORNO' AND prev_event = 'VERKAUF')
    OR (event = 'AKTIVIERUNG' AND NVL (follow_event_event, '') <> 'AKTIVIERUNG')
    OR (event = 'AKTIVIERUNG' AND follow_event = 'DEAKTIVIERUNG')
    OR (event = 'DEAKTIVIERUNG' AND prev_event = 'AKTIVIERUNG')
```

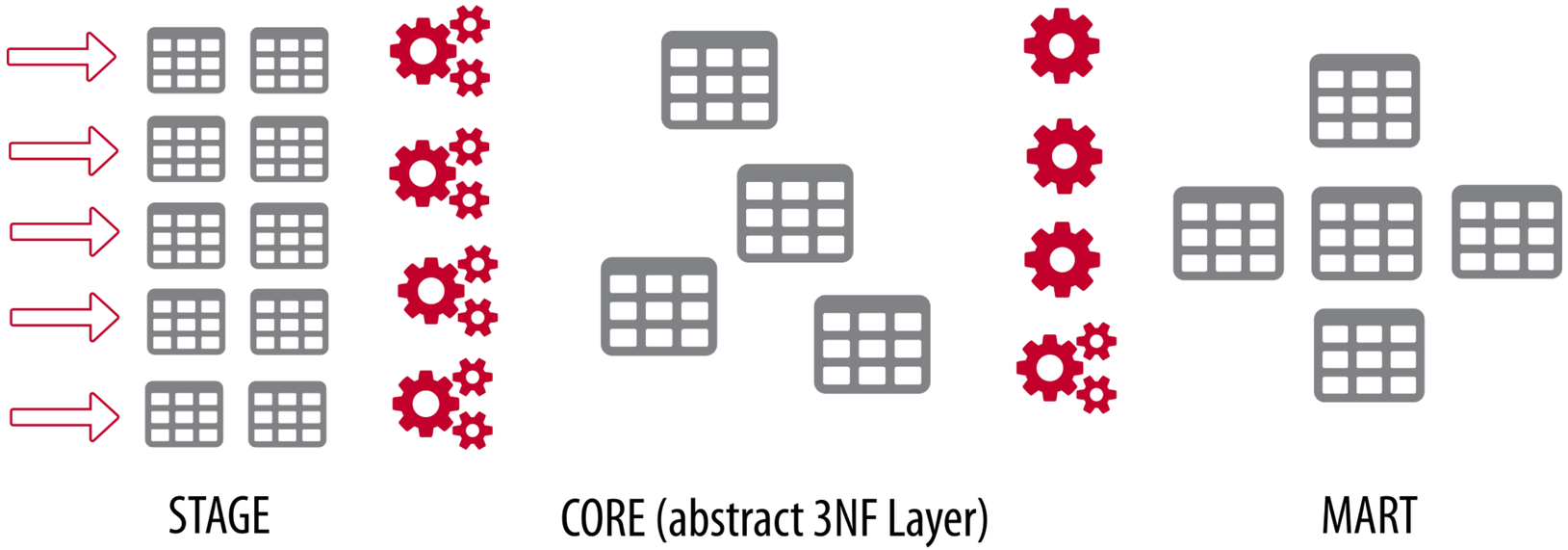
© 2018 SoftQuadrat GmbH

noETL yesSQL in modernen DWHs

© 2018 SoftQuadrat GmbH

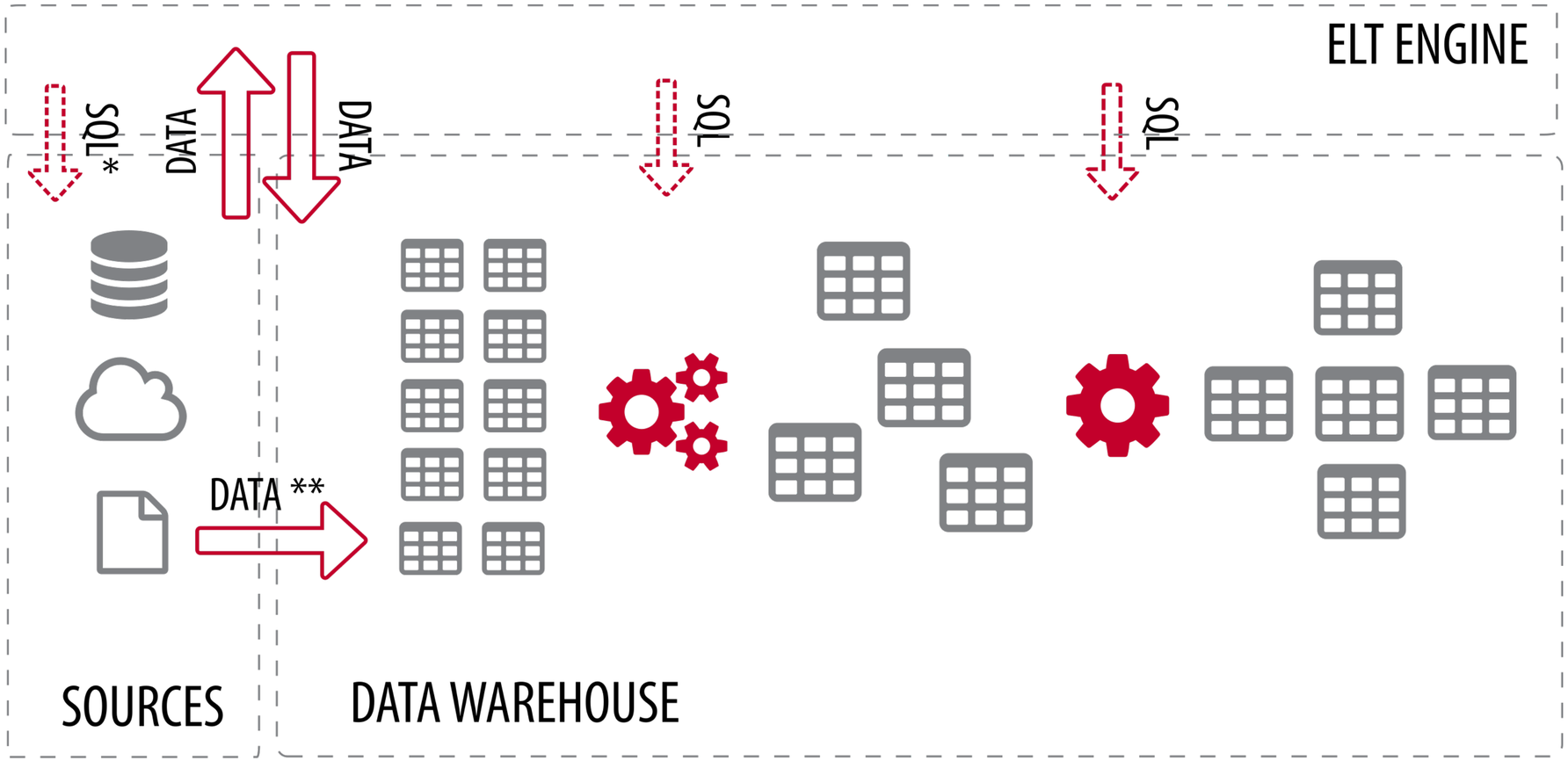
Kontext Data Warehouse

- DWHs sind in einer oder wenigen DBs angesiedelt
- DWHs sind massendatenorientiert
- DWHs benötigen komplexe Transformationen
- und viele Transformationsschritte



© 2018 SoftQuadrat GmbH

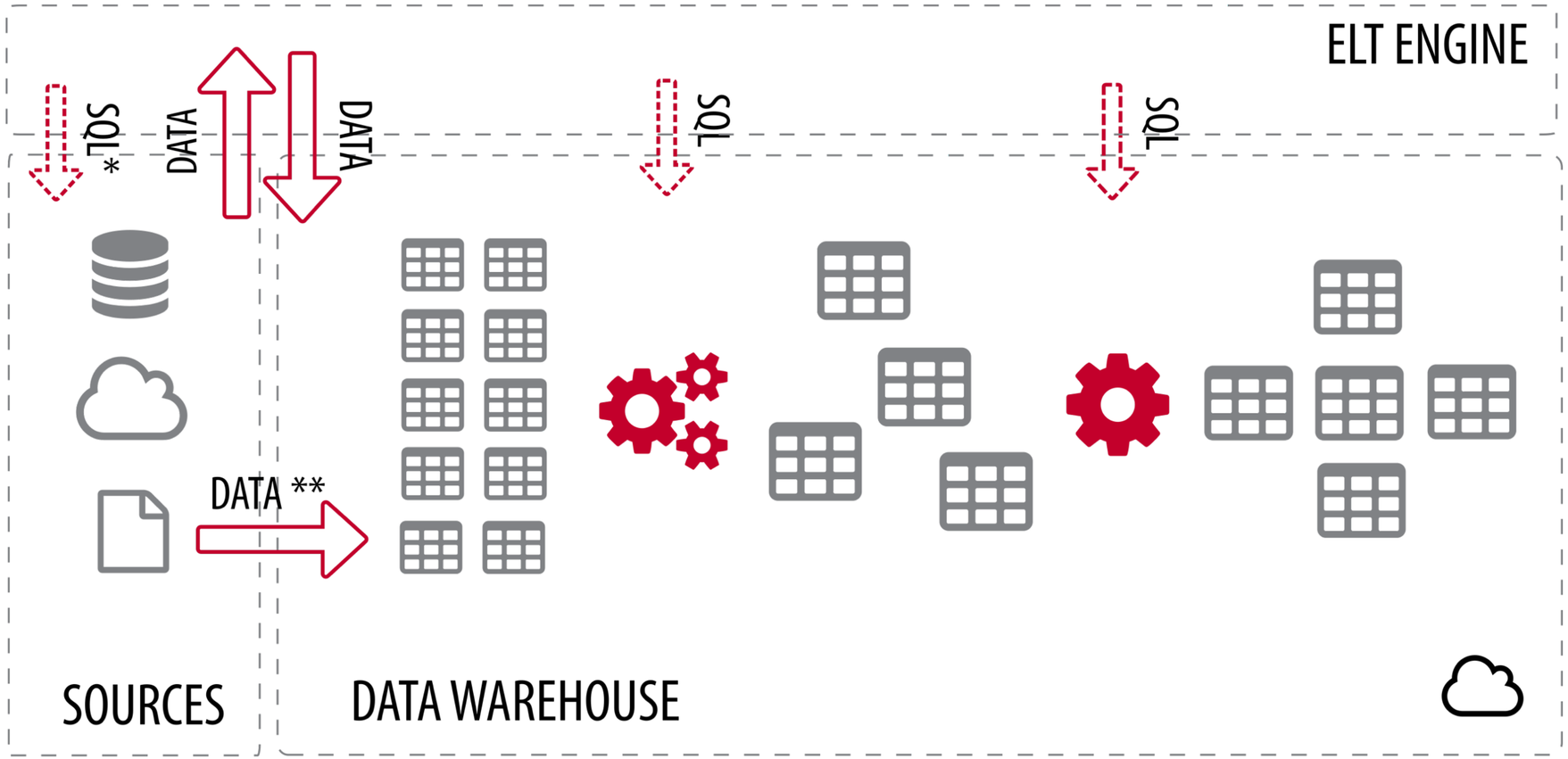
Transformationslandschaft



* SQL oder Source-native Transformationssprache oder OS-Kommando, ** DB-native Intregrationskomponenten

© 2018 SoftQuadrat GmbH

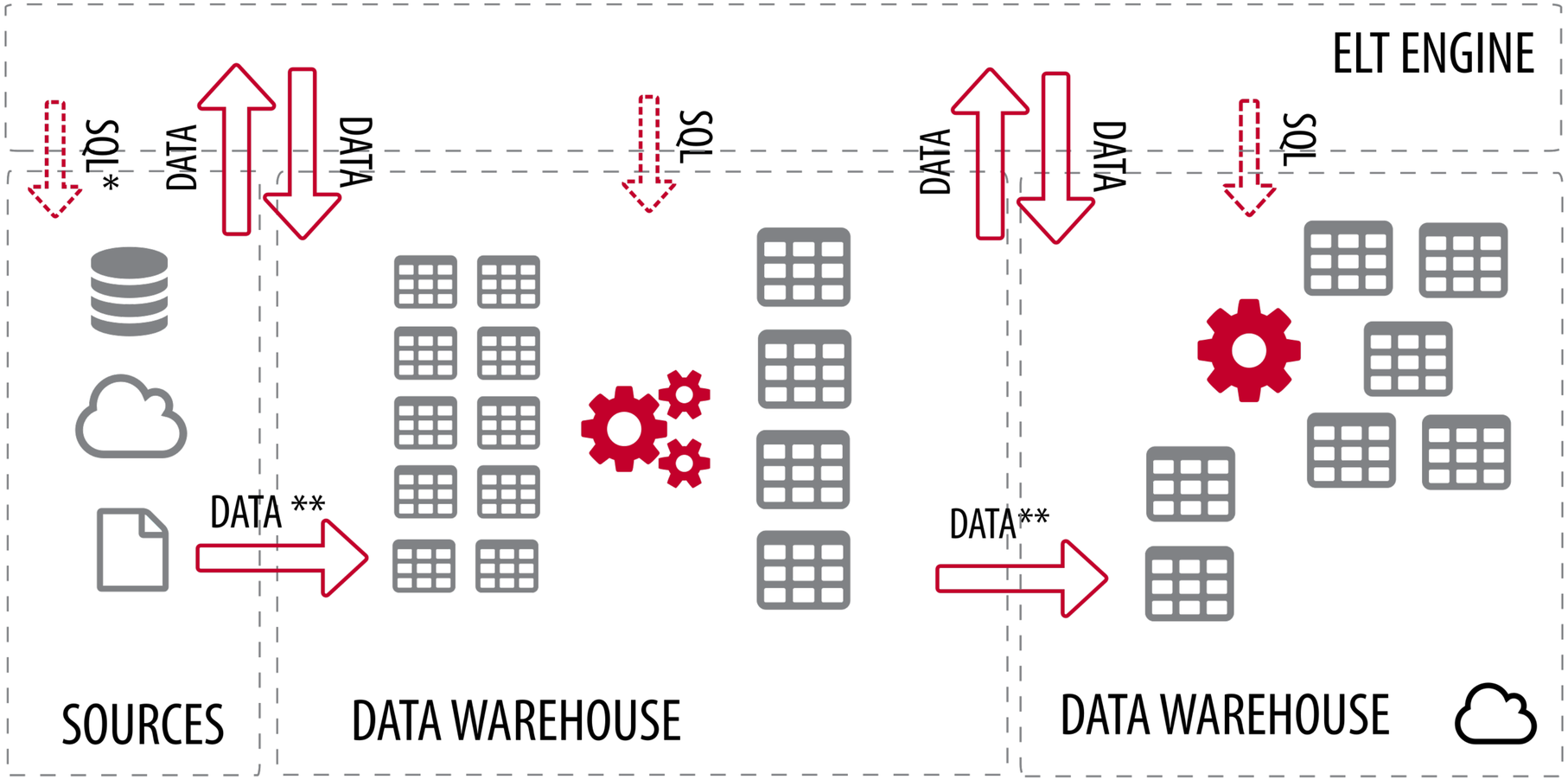
Transformationslandschaft - Cloud



* SQL oder Source-native Transformationssprache oder OS-Kommando, ** DB-native Intregrationskomponenten

© 2018 SoftQuadrat GmbH

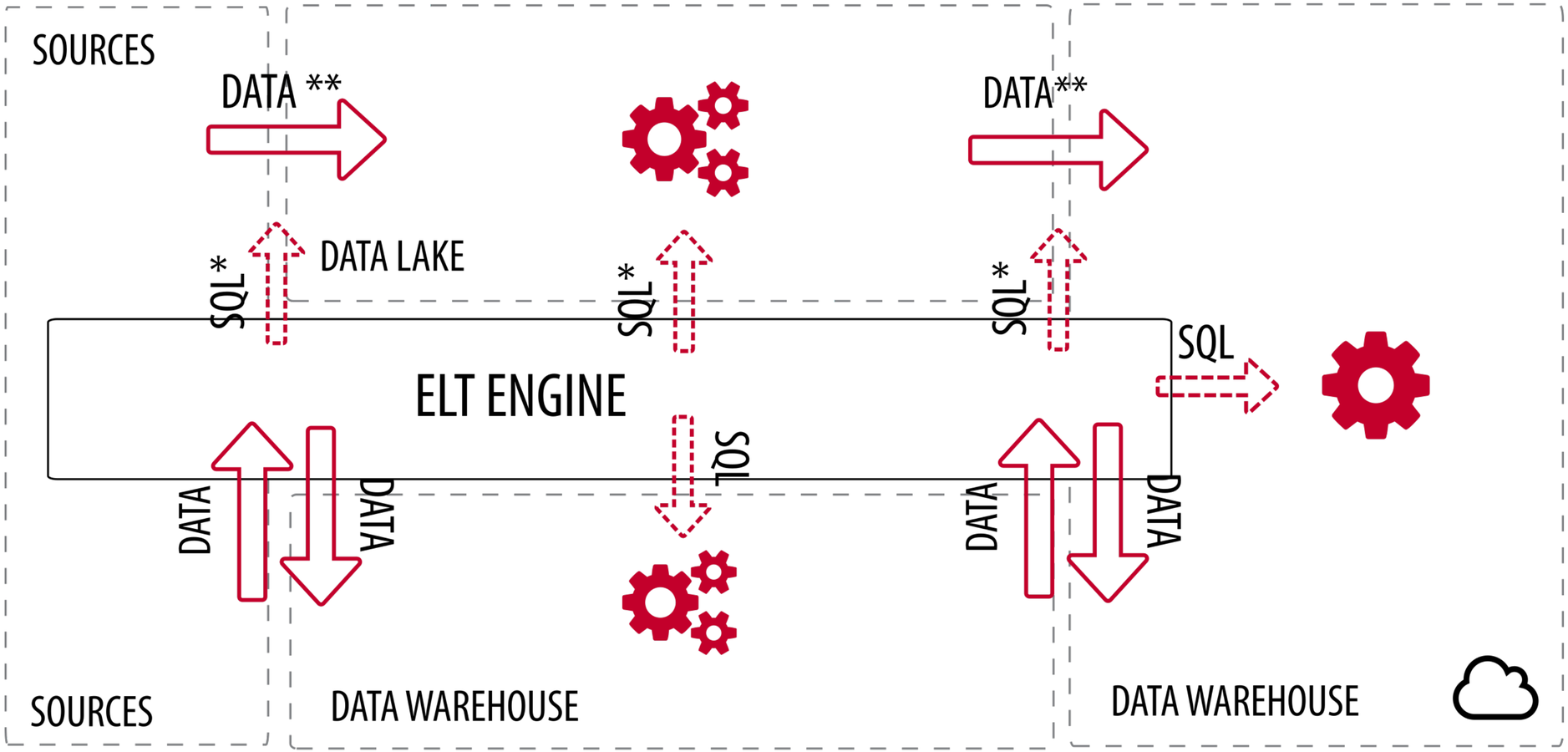
Hybride Transformationslandschaft



* SQL oder Source-native Transformationssprache oder OS-Kommando, ** DB-native Intregrationskomponenten

© 2018 SoftQuadrat GmbH

Hybride Transformationslandschaft mit Data Lake



* SQL oder native Transformationssprache oder OS-Kommando, ** DB-native Intregrationskomponenten

© 2018 SoftQuadrat GmbH

Best Practices: ELT im DWH

- Daten 1:1 stagen, Transformationen erst anschließend durchführen
- Transformationen innerhalb des DWHs runter propagieren (Push down)
- Die Stärken der beteiligten Systemen ausnutzen (DB, MPP DB, In-Memory DB, Hadoop, Spark)
- Wenn sinnvoll, native Integrationsfeatures verwenden
- Wenn sinnvoll, native Parsing-Features (CSV-, JSON-, XML-Parser) verwenden

© 2018 SoftQuadrat GmbH

Best Practices: SQL im DWH

- Möglichst Plain SQL für Transformationslogik verwenden (kein PL/SQL o.ä.)
  - Performance
  - Data Lineage
- SQL-Transformationen möglichst granular halten
- Wiederkehrende Operationen möglichst modularisieren (mit Hilfe des Tools oder Scripting)
- Code Generierung vs Templating vs Modularisierung

© 2018 SoftQuadrat GmbH

Best Practices: DWH Landschaft

- Datenintegration durchdenken:
  - Native Integrationsfeatures
  - ETL Tool
  - Extra Integrationstool
- Richtige Orchestrierung sicherstellen:
  - Externe Workflow Engine/Scheduler
  - ETL Tool
- Metadaten
- Richtige Datenmodellierung:
  - Schichtenmodell vs Short-Path
  - Data Lake
  - Abstraktionsschicht
  - Welche Schichten persistiert man?

© 2018 SoftQuadrat GmbH

Summary

- ELT eignet sich besonders gut für Data Warehousing
- Datenbanken bieten eine passende Platform für ELT
- SQL ist eine flexible, standardisierte und zukunftssichere Sprache für Datenverarbeitung
- noETL yesSQL heißt nicht unbedingt anderes Tooling, sondern andere Denkweise
- Einstieg in noETL yesSQL auch schrittweise ohne ‚Big Bang' möglich (und sinnvoll)

© 2018 SoftQuadrat GmbH

Vielen Dank!

Alec Shalashou
Consultant Data Warehousing
alec@datasqill.de
datasqill.de