EXHIBIT B

Method for enforcing a set of constraints that governs the integrity of information stored in a database system, the constraints being stored in a conceptual rules module in the form of rules for prescribing permitted states and transitions that the database can undertake, the method comprising the steps of delaying constraint checks until the end of a transaction by creating a check stack during the course of the transaction and executing entries on the check stack at the end of the transaction, by a stack maker module operatively connected to a runtime module in said database system: receiving data from said runtime module, and creating and updating said check stack, and retrieving constraints from said conceptual rules module, wherein the check stack contains a list of functions that have to be executed at the end of the transaction, said functions originating from Insert, Delete and Update Data Manipulation Language (DML) operations calling up the stack maker module, the Insert DML operation calling up the stack maker module leading to an insert process being performed on the check stack, the insert process involving placing all checks that have to be executed as a result of an occurrence of a table type being inserted and corresponding conceptual rules being identified for the table type being inserted, the Delete DML operation calling up the stack maker module leading to a delete process being performed on the check stack, the delete process involving removing previously inserted entries on the check stack for the occurrence to be deleted and placing all checks that have to be executed as a result of a table type being deleted and corresponding conceptual rules being identified for the table type being deleted, and the Update DML operation calling up the stack maker module leading to said delete process followed by said insert process being performed on the check stack, and by an enforcer module: receiving check data from the check stack, processing the check data received from the check stack, and providing resulting data to the runtime module, wherein said constraints are constraints executed within the transaction which allow conceptual rules to be broken during the transaction, but allow the database system to be in a consistent state at the beginning and end of the transaction.	Data Quality with ODI: With an approach based on declarative rules, **Oracle Data appropriate tool to help you build a data quality framew inconsistencies. Oracle Data Integrator uses declarativ in its centralized metadata repository. These rules are guarantee the integrity and consistency of enterprise i** benefits add to the overall Data Quality initiative and facilita future business processes addressing this particular need. **Oracle Data Integrator automatically retrieves existing (such as database constraints) by a reverse-engineerin** developers to define additional, user-defined declarative ru data discovery and profiling within ODI, and immediately ch Oracle Data Integrator provides a built-in framework to che ways: ■ Check data in your data servers, to validate that this data rules declared on the datastores in Oracle Data Integrator. a static check and is performed on data models and datast you to profile the quality of the data against rules that are n technology. ■ **Check data while it is moved and transformed by a m checks the data flow against the rules defined on the ta check, correct data can be integrated into the target da automatically moved into error tables.** Source: https://docs.oracle.com/middleware/1212/odi/ODIDG.pdf

Analyst Note: Evidence demonstrates that Oracle Data Integrator (target product) includes a data integrity framework for ensuring the quality of a dat uses data integrity rules (set of constraints) defined in its centralized metadata repository. These rules are applied to application data to guarantee the system. Based on the constraints rules check on data, the correct data can be integrated into target datastore (permitted state) while incorrect data is

US7502791 – Claim 4 as applied to Oracle® Fusion Middleware – Oracle D

Method for enforcing a set of constraints that governs the integrity of information stored in a database system, the constraints being stored in a conceptual rules module in the form of rules for prescribing permitted states and transitions that the database can undertake, the method comprising the steps of

delaying constraint checks until the end of a transaction by creating a check stack during the course of the transaction and executing entries on the check stack at the end of the transaction,

by a stack maker module operatively connected to a runtime module in said database system: receiving data from said runtime module, and

creating and updating said check stack, and retrieving constraints from said conceptual rules module,

wherein the check stack contains a list of functions that have to be executed at the end of the transaction, said functions originating from Insert, Delete and Update Data Manipulation Language (DML) operations calling up the stack maker module,

the Insert DML operation calling up the stack maker module leading to an insert process being performed on the check stack,
the insert process involving placing all checks that have to be executed as a result of an occurrence of a table type being inserted and corresponding conceptual rules being identified for the table type being inserted,

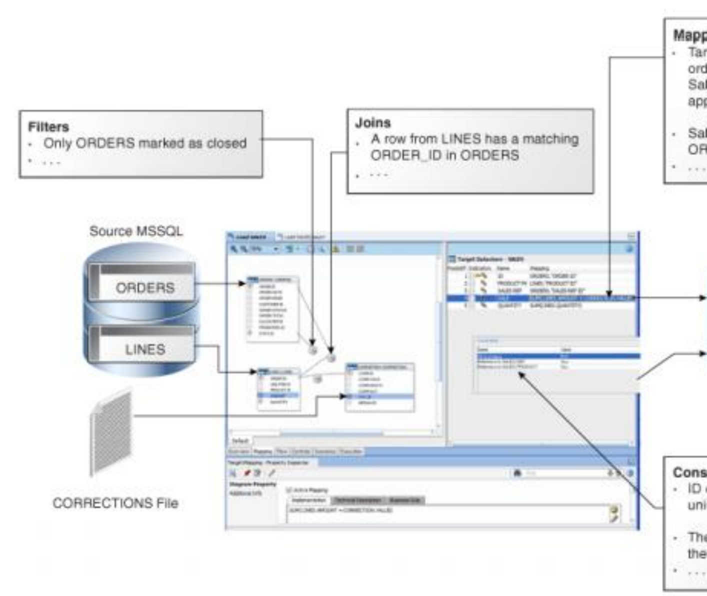
the Delete DML operation calling up the stack maker module leading to a delete process being performed on the check stack,
the delete process involving removing previously inserted entries on the check stack for the occurrence to be deleted and placing all checks that have to be executed as a result of a table type being deleted and corresponding conceptual rules being identified for the table type being deleted, and

the Update DML operation calling up the stack maker module leading to said delete process followed by said insert process being performed on the check stack, and

by an enforcer module: receiving check data from the check stack, processing the check data received from the check stack, and providing resulting data to the runtime module,
wherein said constraints are constraints executed within the transaction which allow conceptual rules to be broken during the transaction, but allow the database system to be in a consistent state at the beginning and end of the transaction.

Introduction to Mappings
A mapping is an Oracle Data Integrator object stored that e
datastores with data transformed from source datastores, **b**
implemented as joins, filters and **constraints.**
A constraint is an object that defines the rules enforced
constraint ensures the validity of the data in a given dat
data of a model. Constraints on the target are used to c
before integration in the target.

Figure 1–3 Implementation using Oracle Data Integrator



Analyst Note: Evidence demonstrates the implementation using Oracle Data Integrator uses a set of constraints (rules) to check the validity (integrity

US7502791 – Claim 4 as applied to Oracle® Fusion Middleware – Oracle D

Method for enforcing a set of constraints that governs the integrity of information stored in a database system, the constraints being stored in a conceptual rules module in the form of rules for prescribing permitted states and transitions that the database can undertake, the method comprising the steps of **delaying constraint checks until the end of a transaction by creating a check stack during the course of the transaction and executing entries on the check stack at the end of the transaction,** by a stack maker module operatively connected to a runtime module in said database system: receiving data from said runtime module, and creating and updating said check stack, and retrieving constraints from said conceptual rules module, wherein the check stack contains a list of functions that have to be executed at the end of the transaction, said functions originating from Insert, Delete and Update Data Manipulation Language (DML) operations calling up the stack maker module, the Insert DML operation calling up the stack maker module leading to an insert process being performed on the check stack, the insert process involving placing all checks that have to be executed as a result of an occurrence of a table type being inserted and corresponding conceptual rules being identified for the table type being inserted, the Delete DML operation calling up the stack maker module leading to a delete process being performed on the check stack, the delete process involving removing previously inserted entries on the check stack for the occurrence to be deleted and placing all checks that have to be executed as a result of a table type being deleted and corresponding conceptual rules being identified for the table type being deleted, and the Update DML operation calling up the stack maker module leading to said delete process followed by said insert process being performed on the check stack, and by an enforcer module: receiving check data from the check stack, processing the check data received from the check stack, and providing resulting data to the runtime module, wherein said constraints are constraints executed within the transaction which allow conceptual rules to be broken during the transaction, but allow the database system to be in a consistent state at the beginning and end of the transaction.	**Check Knowledge Modules (CKM)** **The CKM is in charge of checking that records of a data defined constraints.** The CKM is used to maintain data int overall data quality initiative. The CKM can be used in 2 wa ■ To check the consistency of existing data. This can be do interfaces, by setting the STATIC_CONTROL option to "Yes checked is the data currently in the datastore. In the second datastore is checked after it is loaded. ■ **To check consistency of the incoming data before loa datastore. This is done by using the FLOW_CONTROL o simulates the constraints of the target datastore on the writing to the target.** **The CKM accepts a set of constraints and the name of t** an "E$" error table which it writes all the rejected records to the erroneous records from the checked result set. Source: https://docs.oracle.com/cd/E17904_01/integrate.1111/e12645.pdf **Source/Target Transaction: Transaction where the comm** Source: https://docs.oracle.com/middleware/1212/odi/ODIDG.pdf

Analyst Note: Evidence demonstrates that target product uses check knowledge modules (CKM) to check the consistency of data stored in the datab constraint check process is delayed) against constraints (rules), target product first creates the I$" flow table (check stack) and then CKM accepts a se flow table (check stack) at the end of a transaction. This further illustrates that delaying constraint checks against the data after creating the table as d next slide.

US7502791 – Claim 4 as applied to Oracle® Fusion Middleware – Oracle D

Method for enforcing a set of constraints that governs the integrity of information stored in a database system, the constraints being stored in a conceptual rules module in the form of rules for prescribing permitted states and transitions that the database can undertake, the method comprising the steps of

delaying constraint checks until the end of a transaction by creating a check stack during the course of the transaction and executing entries on the check stack at the end of the transaction,

by a stack maker module operatively connected to a runtime module in said database system: receiving data from said runtime module, and

creating and updating said check stack, and retrieving constraints from said conceptual rules module,

wherein the check stack contains a list of functions that have to be executed at the end of the transaction, said functions originating from Insert, Delete and Update Data Manipulation Language (DML) operations calling up the stack maker module,

the Insert DML operation calling up the stack maker module leading to an insert process being performed on the check stack,
the insert process involving placing all checks that have to be executed as a result of an occurrence of a table type being inserted and corresponding conceptual rules being identified for the table type being inserted,

the Delete DML operation calling up the stack maker module leading to a delete process being performed on the check stack,
the delete process involving removing previously inserted entries on the check stack for the occurrence to be deleted and placing all checks that have to be executed as a result of a table type being deleted and corresponding conceptual rules being identified for the table type being deleted, and

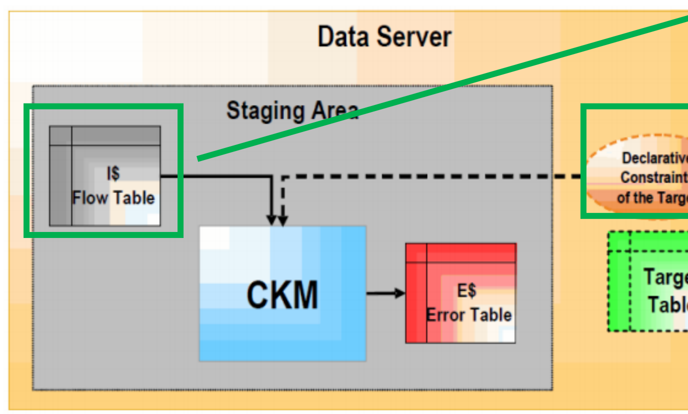
the Update DML operation calling up the stack maker module leading to said delete process followed by said insert process being performed on the check stack, and

by an enforcer module: receiving check data from the check stack, processing the check data received from the check stack, and providing resulting data to the runtime module,

wherein said constraints are constraints executed within the transaction which allow conceptual rules to be broken during the transaction, but allow the database system to be in a consistent state at the beginning and end of the transaction.

The following figures show how a CKM operates in both ST
FLOW_CONTROL modes.
In FLOW_CONTROL mode, the CKM reads the constrai
Interface. It checks these constraints against the data conta
the staging area. Records that violate these constraints are
staging area.

Figure 1–3 Check Knowledge Module (FLOW_CONTROL)



Source: https://docs.oracle.com/cd/E17904_01/integrate.1111/e12645.pdf

Analyst Note: The screenshot demonstrates the check knowledge module uses constraint rules against the "I$" flow table (check stack) that governs
database system.

US7502791 – Claim 4 as applied to Oracle® Fusion Middleware – Oracle D

DOCKET ALARM

Explore Litigation Insights

Docket Alarm provides insights to develop a more informed litigation strategy and the peace of mind of knowing you're on top of things.

Real-Time Litigation Alerts

Keep your litigation team up-to-date with **real-time alerts** and advanced team management tools built for the enterprise, all while greatly reducing PACER spend.

Our comprehensive service means we can handle Federal, State, and Administrative courts across the country.

Advanced Docket Research

With over 230 million records, Docket Alarm's cloud-native docket research platform finds what other services can't. Coverage includes Federal, State, plus PTAB, TTAB, ITC and NLRB decisions, all in one place.

Identify arguments that have been successful in the past with full text, pinpoint searching. Link to case law cited within any court document via Fastcase.

Analytics At Your Fingertips

Learn what happened the last time a particular judge, opposing counsel or company faced cases similar to yours.

Advanced out-of-the-box PTAB and TTAB analytics are always at your fingertips.

API

Docket Alarm offers a powerful API (application programming interface) to developers that want to integrate case filings into their apps.

LAW FIRMS

Build custom dashboards for your attorneys and clients with live data direct from the court.

Automate many repetitive legal tasks like conflict checks, document management, and marketing.

FINANCIAL INSTITUTIONS

Litigation and bankruptcy checks for companies and debtors.

E-DISCOVERY AND LEGAL VENDORS

Sync your system to PACER to automate legal marketing.

WHAT WILL YOU BUILD? | **sales@docketalarm.com** | **1-866-77-FASTCASE**