# A column-level data lineage model for describing a database's dataflow

(James Wang at Gudu software https://www.gudusoft.com)

### teaser

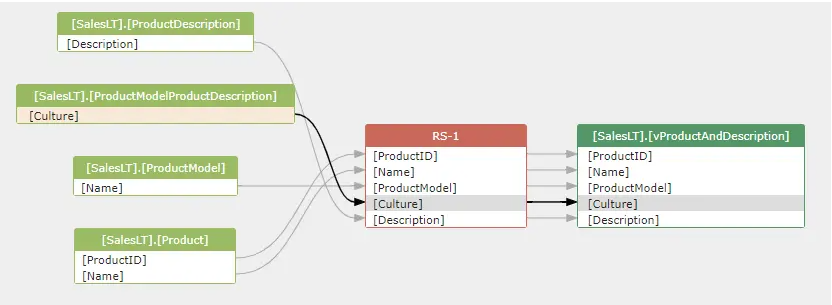
*Data lineage is a map of the data journey, which includes its origin, each stop along the way, and an explanation on how and why the data has moved over time.  It includes all transformations the data underwent along the way—how the data was transformed, what changed, and why.*

*We offer a column-level lineage model to uniformly define data lineage. This methodology makes it possible to collect lineage metadata consistently, resulting in a better knowledge of how data is created and used.*

### Body

This data lineage model is primarily concerned with the data lineage produced by the SQL scripts of different database types, but it can easily enhanced to keep metadata about datasets, jobs, and runs.

The entity in the data lineage model includes dataset such as table, column, function, relation and other entities . The combination of the entity and dataflow shows the lineage from one table/column to another.



**1. A dataflow unit**

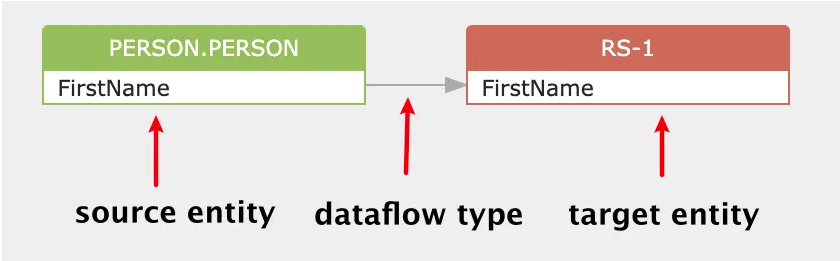
A source entity, a target entity, and the dataflow type that connects them all together make up a dataflow unit.

SELECT p.FirstName

from Person.Person AS p

This is the dataflow generated for the above SQL query.

***person.persion.FirstName -> direct -> RS-1.FirstName***



**1.1 Source, target entity**

Source and target entity usually refers to dataset such as table, view and other relations such as common table expression, result set generated during the execution of the query. It may also refer to a file in the HDFS system and etc.

**1.2 Dataflow type**

There are two types of dataflow: direct and indirect.

**1.2.1 Direct dataflow**

The direct dataflow means the data of the target entity comes directly from the source entity.

In the above diagram, the data of RS-1.FirstName comes from the Person.FirstName directly.

An arrow is used to represent a direct dataflow in the diagram:



**1.2.2 Indirect dataflow**

The indirect dataflow means the data of the target column is not comes from the source column, but the data of the source column/table impact the result data of the target column.

A dotted line arrow is used to represent an indirect dataflow in the diagram:



The source column in the indirect dataflow usually appears in the following clause:

* Where clause
* Group by clause
* Winddows function
* Join condition

SELECT deptno, COUNT() num\_emp, SUM(SAL) sal\_sum

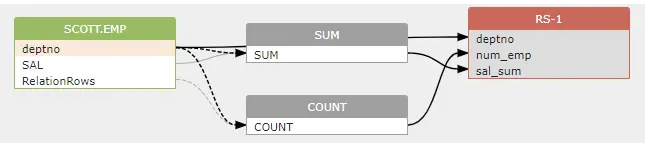
FROM scott.emp

GROUP BY deptno

The value of COUNT() and SUM(SAL) is impacted by the value of column deptno in the group by clause. So the indirect dataflows will be created like this:

***scott.emp.deptno -> indirect -> COUNT()***

***scott.emp.deptno -> indirect -> SUM(SAL)***



**2. The entity in dataflow**

When build dataflow between 2 entities: the source and target entity. They can be column to column, or, table to colum, or table to table.

**2.1 Column to column**

This is the most often cases. Both entities in a dataflow are columns.

**2.2 Table to column**

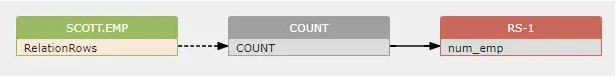
When we say a table impact the value of a column, we usually means the total number of rows of a table impact the value of a column, usually, this column is derived from a COUNT() function.

SELECT COUNT() num\_emp

FROM scott.emp

A table to column dataflow is represented by using a RelationRows pseduo column. This build an indirect dataflow from scott.emp.RelationRows to RS-1.num\_emp

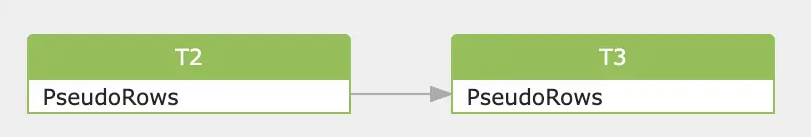
***scott.emp.RelationRows -> indirect -> COUNT() -> RS-1.num\_emp***



**2.3 Table to table**

Sometimes, there will be a dataflow between 2 tables. For example, in an alter table rename SQL statement; a table to table dataflow will be created. Actually, a table to table dataflow is represented by a column to column dataflow using the RelationRows pseudo column.

alter table t2 rename to t3;



**3. Data lineage**

A data lineage consists of lots of basic dataflow units.

CREATE VIEW vsal

AS

SELECT a.deptno "Department",

a.num\_emp / b.total\_count "Employees",

a.sal\_sum / b.total\_sal "Salary"

FROM (SELECT deptno,

Count() num\_emp,

SUM(sal) sal\_sum

FROM scott.emp

WHERE city = 'NYC'

GROUP BY deptno) a,

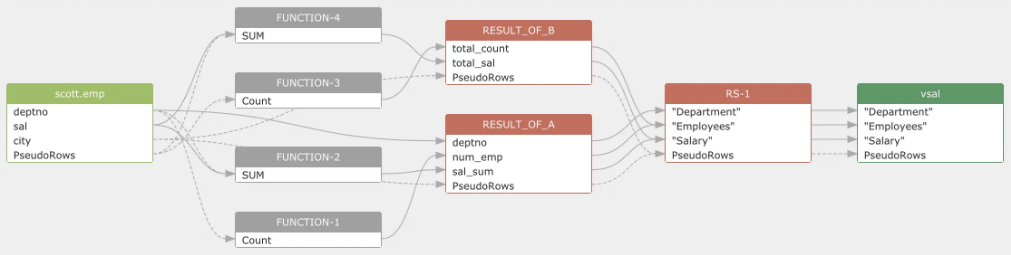
(SELECT Count() total\_count,

SUM(sal) total\_sal

FROM scott.emp

WHERE city = 'NYC') b

The data lineage diagram:



The output is additionally offered in XML or JSON formats. Please [check this document](https://github.com/sqlparser/sqlflow_public/blob/master/doc/data-lineage-format/data-lineage-format-reference.md) for the detail.

**4. Data transformation**

Calculation compliance users need an exact derivation logic extraction implemented to clearly see what derivations have been applied to attributes in order to provide a calculation.

This data lineage model exposes the transformations so that user can utilize that to determine the derivation logic for a particular column. This will be a relevant section of SQL and not the entire stored procedure or SQL file.

INSERT INTO deptsal

(dept\_no,

dept\_name,

salary)

SELECT d.deptno,

d.dname,

SUM(e.sal + Nvl(e.comm, 0)) AS sal

FROM dept d

left join (SELECT \*

FROM emp

WHERE hiredate > DATE '1980-01-01') e

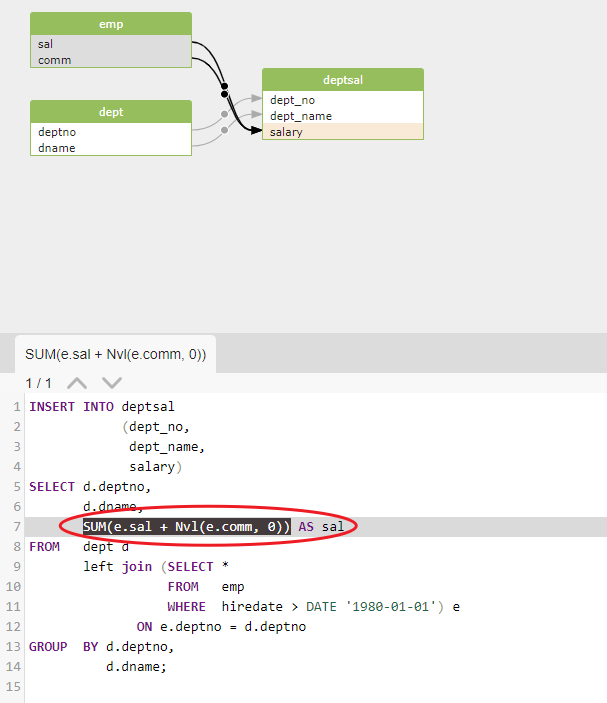
ON e.deptno = d.deptno

GROUP BY d.deptno,

d.dname;

In the above SQL, data transformation can be modeled like this:

***Emp.sal -> data transform (SUM(e.sal + Nvl(e.comm, 0))) -> deptsal.salary***

**

**5. Implementation of the data lineage model**

This column-level data lineage model was applied by Gudu Software Company in its [Gudu SQLFlow product](https://www.gudusoft.com/sql-data-lineage-tool/), an automated data lineage tool that focuses on processing SQL queries to find and visualize data lineage that demonstrates the entire data cycle.

**6. about the Author:**

James Wang is a Senior Technical Evangelist at Gudu software.

With more than 10 years of experience in data infrastructure and applications, James has a proven track record of introducing new technologies to users and business analysts.