(19) 대한민국특허청(KR)
(12) 등록특허공보(B1)

(45) 공고일자 2022년06월15일
(11) 등록번호 10-2409552
(24) 등록일자 2022년06월13일

(51) 국제특허분류(Int. Cl.)
G06F 16/00 (2019.01) *G06F 13/10* (2018.01)
G06Q 10/06 (2012.01)
(52) CPC특허분류
G06F 16/86 (2019.01)
G06F 13/10 (2013.01)
(21) 출원번호 10-2016-7028170
(22) 출원일자(국제) 2015년03월16일
심사청구일자 2020년02월05일
(85) 번역문제출일자 2016년10월11일
(65) 공개번호 10-2016-0132942
(43) 공개일자 2016년11월21일
(86) 국제출원번호 PCT/US2015/020660
(87) 국제공개번호 WO 2015/139018
국제공개일자 2015년09월17일
(30) 우선권주장
61/953,021 2014년03월14일 미국(US)
(56) 선행기술조사문헌
US20140032617 A1
US06035300 A

(73) 특허권자
아브 이니티오 테크놀로지 엘엘시
미국 02421 매사추세츠주 렉싱턴 스프링 스트리트 201
(72) 발명자
로버츠, 제드
미국 메사추세츠 02461-2024 뉴턴 리우드 로드 7
스탠필, 크레이그 더블유.
미국 메사추세츠 01773 링컨 허클베리 힐 로드 43
스튜더, 스콧
미국 메사추세츠 01833 조지타운 필스버리 레인 27
(74) 대리인
인비전 특허법인

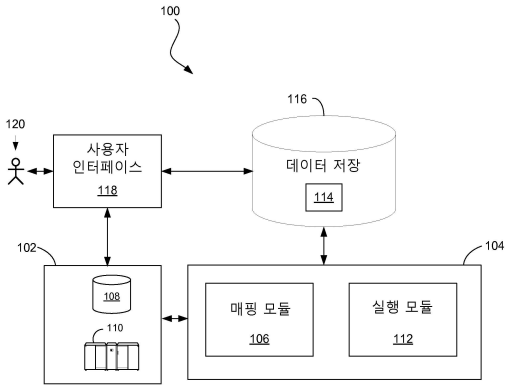
전체 청구항 수 : 총 19 항 심사관 : 안지현

(54) 발명의 명칭 **키드 엔티티들의 속성 매핑**

(57) 요 약

하나 이상의 매핑들(304) 각각은 입력 엔티티의 입력 속성들과 출력 엔티티의 출력 속성들 사이의 대응 관계를 정의하고, 상기 입력과 출력 엔티티들 각각은 고유 키의 일부로서 식별된 하나 이상의 키 속성들을 포함한다. 사용자 인터페이스에 디스플레이되는 결과 정보를 계산하는 것은 제1 출력 엔티티의 인스턴스들을 생성하기 위해 제1 입력 엔티티의 인스턴스들을 처리하는 것, 상기 제1 출력 엔티티의 키 속성들 각각에 대응하는 상기 제1 입력 엔티티의 하나 이상의 매핑된 입력 속성들을 결정하는 것; 상기 결정된 하나 이상의 매핑된 입력 속성들에 기반하여 상기 제1 출력 엔티티의 인스턴스들을 생성하는 것; 처리된 상기 제1 입력 엔티티의 인스턴스들의 총 수(316A)를 계산하는 것; 및 생성된 상기 제1 출력 엔티티의 인스턴스들의 총 수(316B)를 계산하는 것을 포함한다.

대 표 도



(52) CPC특허분류
G06F 16/2455 (2019.01)
G06F 16/258 (2019.01)
G06F 16/282 (2019.01)
G06F 16/288 (2019.01)
G06Q 10/067 (2013.01)

명 세 서

청구범위

청구항 1

컴퓨팅 시스템으로서, 상기 시스템은

적어도 하나의 속성들을 가지는 각각의 엔티티로 복수의 엔티티들을 나타내는 엔티티 데이터를 저장하는 데이터 저장 시스템 - 상기 엔티티들 중 적어도 일부 각각은 여러 인스턴스들을 가지고, 상기 인스턴스들 중 적어도 일부 각각은 상기 속성들 중 적어도 하나에 대한 각각의 값들을 가지며, 상기 복수의 엔티티들은 복수의 입력 엔티티들과 복수의 출력 엔티티들을 포함함 - ;

각각이 상기 입력 엔티티들 중 하나의 적어도 하나의 입력 속성들과 상기 출력 엔티티들 중 하나의 적어도 하나의 출력 속성들 사이의 대응관계를 정의하는 적어도 하나의 매핑들을 포함하는 입력 데이터를 수신하기 위한 입력 디바이스 또는 포트 - 상기 입력 엔티티는 상기 입력 엔티티에 대한 고유 키의 일부로서 식별된 하나 이상의 키 속성들을 포함하고, 상기 출력 엔티티는 상기 출력 엔티티에 대한 고유 키의 일부로서 식별된 적어도 하나의 키 속성들을 포함함 - ;

상기 입력 데이터를 수신하도록 구성된 사용자 인터페이스를 디스플레이하기 위한 출력 디바이스 또는 포트; 및

상기 사용자 인터페이스에 디스플레이되는 결과 정보를 계산하도록 구성된 적어도 하나의 프로세서 - 상기 결과 정보는 상기 입력 데이터에 포함된 상기 적어도 하나의 매핑들에 따라 상기 출력 엔티티들의 인스턴스들을 생성하기 위해 상기 입력 엔티티들의 인스턴스들을 처리한 결과를 특성화함- 를 포함하되, 상기 계산은

제1 출력 엔티티의 인스턴스들을 생성하기 위해 제1 입력 엔티티의 인스턴스들을 처리하는 것;

상기 적어도 하나의 매핑들에 기반하여, 상기 제1 출력 엔티티의 적어도 하나의 키 속성들 각각에 대응하는 상기 제1 입력 엔티티의 적어도 하나의 매핑된 입력 속성들을 결정하는 것;

상기 결정된 적어도 하나의 매핑된 입력 속성들에 기반하여 상기 제1 출력 엔티티의 인스턴스들을 생성하는 것;

처리된 상기 제1 입력 엔티티의 인스턴스들의 총 수를 계산하는 것; 및

생성된 상기 제1 출력 엔티티의 인스턴스들의 총 수를 계산하는 것을 포함하는, 컴퓨팅 시스템.

청구항 2

제1항에 있어서,

상기 사용자 인터페이스에 상기 결과 정보를 디스플레이하는 것은 상기 제1 출력 엔티티의 표현과 관련하여 상기 제1 출력 엔티티의 인스턴스들의 총 수를 디스플레이하는 것을 포함하는, 컴퓨팅 시스템.

청구항 3

제2항에 있어서,

상기 사용자 인터페이스에 상기 결과 정보를 디스플레이하는 것은 상기 제1 입력 엔티티의 표현과 관련하여 상기 제1 입력 엔티티의 인스턴스들의 총 수를 디스플레이하는 것을 포함하는, 컴퓨팅 시스템.

청구항 4

제1항에 있어서,

상기 사용자 인터페이스에 상기 결과 정보를 디스플레이하는 것은 상기 디스플레이된 입력 엔티티의 속성들과 디스플레이된 출력 엔티티의 속성들 사이의 적어도 하나의 매핑들을 나타내는 여러 요소들을 디스플레이하는 것, 상기 디스플레이된 입력 엔티티와 디스플레이된 출력 엔티티 사이의 임의의 매핑들에 대한 입력 데이터가 출력 속성을 (1) 동일한 이름의 입력 속성, 또는 (2) 상수 값으로 할당하는지 여부를 표시하는 각각의 요소에 대한 아이콘을 디스플레이하는 것을 포함하는, 컴퓨팅 시스템.

청구항 5

제1항에 있어서,

상기 출력 엔티티의 적어도 하나의 키 속성들 각각에 대응하는 상기 입력 엔티티의 적어도 하나의 매핑된 입력 속성들을 결정하는 것은 상기 적어도 하나의 매핑된 입력 속성들이 상기 출력 엔티티의 각각의 키 속성들과 일대일 대응관계를 가지는지 여부를 결정하는 것을 포함하는, 컴퓨팅 시스템.

청구항 6

제1항에 있어서,

상기 계산은 상기 매핑된 입력 속성들이 (1) 상기 입력 엔티티의 키 속성들 모두, 또는 (2) 상기 입력 엔티티의 키 속성들 모두보다 적은 것을 포함하는지를 결정하기 위해 상기 매핑된 입력 속성들을 상기 입력 엔티티의 적어도 하나의 키 속성들과 비교하는 것을 더 포함하는, 컴퓨팅 시스템.

청구항 7

제6항에 있어서,

상기 처리는 (1) 상기 매핑된 입력 속성들이 상기 입력 엔티티의 키 속성들 모두를 포함한다는 결정에 응하여, 상기 출력 엔티티의 인스턴스들과 매칭 키 속성들(matching key attribute)을 가지는 상기 입력 엔티티의 인스턴스들 사이의 일대일 대응관계, 또는 (2) 상기 매핑된 입력 속성들이 상기 입력 엔티티의 키 속성들 모두보다 적은 것을 포함한다는 결정에 응하여, 상기 매핑된 입력 속성들에 대해 동일한 값들을 공유하는 상기 입력 엔티티의 여러 인스턴스들의 집계(aggregation)에 기반하여 상기 출력 엔티티의 인스턴스들을 생성하는 것을 더 포함하는, 컴퓨팅 시스템.

청구항 8

제1항에 있어서,

상기 엔티티 데이터는 계층 구조에 따라 관련된 복수의 출력 엔티티들을 나타내고, 적어도 하나의 루트 출력 엔티티(root output entity)는 상기 계층 구조의 최상위 레벨에 있고 적어도 하나의 출력 엔티티들은 상기 계층 구조의 최상위 레벨 아래의 적어도 하나의 레벨들에 있으며, 상기 루트 출력 엔티티보다 낮은 레벨의 각각의 출력 엔티티는 단일 출력 엔티티의 서브 엔티티(sub-entity)인, 컴퓨팅 시스템.

청구항 9

제8항에 있어서,

상기 엔티티 데이터는 계층 구조에 따라 관련된 복수의 입력 엔티티들을 나타내고, 적어도 하나의 루트 입력 엔티티는 상기 계층 구조의 최상위 레벨에 있고 적어도 하나의 입력 엔티티들이 상기 계층 구조의 최상위 레벨 아래의 적어도 하나의 레벨들에 있으며, 상기 루트 입력 엔티티보다 낮은 레벨의 각각의 입력 엔티티는 단일 입력 엔티티의 서브 엔티티인, 컴퓨팅 시스템.

청구항 10

제8항에 있어서,

계층 구조에 따라 관련된 상기 복수의 출력 엔티티들에 관련되지 않은 적어도 제1 엔티티는 상기 입력 데이터에 포함된 상기 매핑들 중 적어도 하나에 의해 출력 속성으로 참조되는 적어도 하나의 속성을 포함하는, 컴퓨팅 시스템.

청구항 11

제10항에 있어서,

상기 제1 엔티티는 상기 입력 데이터에 포함된 상기 매핑들 중 적어도 하나에 의해 입력 속성으로 참조되는 적어도 하나의 속성을 포함하는, 컴퓨팅 시스템.

청구항 12

제1항에 있어서,

제2 엔티티의 서브 엔티티인 제1 엔티티의 복수의 인스턴스들 각각은 상기 제2 엔티티의 특정 인스턴스를 식별하는 상기 제1 엔티티의 키 속성의 공통 값을 포함하는, 컴퓨팅 시스템.

청구항 13

제12항에 있어서,

상기 제1 엔티티는 레코드들의 제1 세트에 대응하고, 상기 제2 엔티티는 레코드들의 제2 세트에 대응하며, 상기 제1 엔티티의 키 속성은 레코드들의 상기 제2 세트의 특정 레코드의 기본 키 필드에 포함된 값을 식별하는 레코드들의 상기 제1 세트의 외래 키 필드에 대응하는, 컴퓨팅 시스템.

청구항 14

제1항에 있어서,

제2 엔티티의 서브 엔티티인 제1 엔티티의 복수의 인스턴스들은 상기 제2 엔티티의 특정 인스턴스의 데이터 구조 내에 포함된 벡터(vector)의 복수의 요소들에 대응하는, 컴퓨팅 시스템.

청구항 15

제14항에 있어서,

상기 처리는 상기 출력 엔티티의 인스턴스들을 생성하기 위해 상기 입력 엔티티의 인스턴스들을 처리하도록 데이터플로 그래프를 이용하여 상기 출력 엔티티의 인스턴스들을 생성하는 것을 더 포함하고, 상기 데이터플로 그래프는 엔티티의 인스턴스들에 연산들을 수행하도록 구성된 컴포넌트들(components)을 나타내는 노드들과 컴포넌트들 사이의 인스턴스들의 흐름들을 나타내는 노드들 사이의 링크들을 포함하는, 컴퓨팅 시스템.

청구항 16

제15항에 있어서,

상기 데이터플로 그래프는 상기 적어도 하나의 매핑들의 입력 속성들에 기반하여 또 다른 엔티티의 인스턴스의 데이터 구조로부터 서브 엔티티의 인스턴스들의 적어도 하나의 벡터들을 추출하도록 구성되는 적어도 하나의 분할 컴포넌트(spilt component)와 상기 적어도 하나의 매핑들의 출력 속성들에 기반하여 또 다른 엔티티의 인스턴스의 데이터 구조로 서브 엔티티의 인스턴스들의 적어도 하나의 벡터들을 삽입하도록 구성된 적어도 하나의 결합 컴포넌트(combine component)를 포함하는, 컴퓨팅 시스템.

청구항 17

컴퓨팅 시스템에 있어서, 상기 시스템은

적어도 하나의 속성들을 가지는 각각의 엔티티로 복수의 엔티티들을 나타내는 엔티티 데이터를 저장하기 위한 수단 - 상기 엔티티들 중 적어도 일부 각각은 여러 인스턴스들을 가지고, 상기 인스턴스들 중 적어도 일부 각각은 상기 속성들 중 적어도 하나에 대한 각각의 값을 가지며, 상기 복수의 엔티티들은 복수의 입력 엔티티들과 복수의 출력 엔티티들을 포함함 - ;

각각이 상기 입력 엔티티들 중 하나의 적어도 하나의 입력 속성들과 상기 출력 엔티티들 중 하나의 적어도 하나의 출력 속성들 사이의 대응관계를 정의하는 적어도 하나의 매핑들을 포함하는 입력 데이터를 수신하기 위한 수단 - 상기 입력 엔티티는 상기 입력 엔티티에 대한 고유 키의 일부로서 식별된 적어도 하나의 키 속성들을 포함하고, 상기 출력 엔티티는 상기 출력 엔티티에 대한 고유 키의 일부로서 식별된 적어도 하나의 키 속성들을 포함함 - ;

상기 입력 데이터를 수신하도록 구성된 사용자 인터페이스를 디스플레이하기 위한 수단; 및

상기 사용자 인터페이스에 디스플레이된 결과 정보를 계산하기 위한 수단 - 상기 결과 정보는 상기 입력 데이터에 포함된 상기 적어도 하나의 매핑들에 따라 상기 출력 엔티티들의 인스턴스들을 생성하기 위해 상기 입력 엔

티티들의 인스턴스들을 처리한 결과를 특성화함 - 을 포함하되, 상기 계산은

제1 출력 엔티티의 인스턴스들을 생성하기 위해 제1 입력 엔티티의 인스턴스들을 처리하는 것;

상기 적어도 하나의 매핑들에 기반하여, 상기 제1 출력 엔티티의 적어도 하나의 키 속성들 각각에 대응하는 상기 제1 입력 엔티티의 적어도 하나의 매핑된 입력 속성들을 결정하는 것;

상기 결정된 적어도 하나의 매핑된 입력 속성들에 기반하여 상기 제1 출력 엔티티의 인스턴스들을 생성하는 것;

처리된 상기 제1 입력 엔티티의 인스턴스들의 총 수를 계산하는 것; 및

생성된 상기 제1 출력 엔티티의 인스턴스들의 총 수를 계산하는 것을 포함하는, 컴퓨팅 시스템.

청구항 18

컴퓨팅 시스템에서 데이터를 처리하기 위한 방법에 있어서, 상기 방법은

데이터 저장 시스템에, 적어도 하나의 속성들을 가지는 각각의 엔티티로 복수의 엔티티들을 나타내는 엔티티 데이터를 저장하는 단계 - 상기 엔티티들 중 적어도 일부 각각은 여러 인스턴스들을 가지고, 상기 인스턴스들 중 적어도 일부 각각은 상기 속성들 중 적어도 하나에 대한 각각의 값들을 가지며, 상기 복수의 엔티티들은 복수의 입력 엔티티들과 복수의 출력 엔티티들을 포함함 - ;

입력 디바이스 또는 포트를 통해, 각각이 상기 입력 엔티티들 중 하나의 적어도 하나의 입력 속성들과 상기 출력 엔티티들 중 하나의 적어도 하나의 출력 속성들 사이의 대응관계를 정의하는 적어도 하나의 매핑들을 포함하는 입력 데이터를 수신하는 단계 - 상기 입력 엔티티는 상기 입력 엔티티에 대한 고유 키의 일부로서 식별된 적어도 하나의 키 속성들을 포함하고, 상기 출력 엔티티는 상기 출력 엔티티에 대한 고유 키의 일부로서 식별된 적어도 하나의 키 속성들을 포함함 -

출력 디바이스 또는 포트를 통해, 상기 입력 데이터를 수신하도록 구성된 사용자 인터페이스를 디스플레이하는 단계; 및

적어도 하나의 프로세서로, 상기 사용자 인터페이스에 디스플레이되는 결과 정보를 계산하는 단계 - 상기 결과 정보는 상기 입력 데이터에 포함된 상기 적어도 하나의 매핑들에 따라 상기 출력 엔티티들의 인스턴스들을 생성하기 위해 상기 입력 엔티티들의 인스턴스들을 처리한 결과를 특성화함 - 를 포함하되, 상기 계산하는 단계는

제1 출력 엔티티의 인스턴스들을 생성하기 위해 제1 입력 엔티티의 인스턴스들을 처리하는 단계;

상기 적어도 하나의 매핑들에 기반하여, 상기 제1 출력 엔티티의 적어도 하나의 키 속성들 각각에 대응하는 상기 제1 입력 엔티티의 적어도 하나의 매핑된 입력 속성들을 결정하는 단계;

상기 결정된 적어도 하나의 매핑된 입력 속성들에 기반하여 상기 제1 출력 엔티티의 인스턴스들을 생성하는 단계;

처리된 상기 제1 입력 엔티티의 인스턴스들의 총 수를 계산하는 단계; 및

생성된 상기 제1 출력 엔티티의 인스턴스들의 총 수를 계산하는 단계를 포함하는, 컴퓨팅 시스템에서 데이터를 처리하기 위한 방법.

청구항 19

컴퓨터 판독 가능한 매체 상에 비 일시적 형태로 저장된 소프트웨어에 있어서, 상기 소프트웨어는 컴퓨팅 시스템이

데이터 저장 시스템에, 적어도 하나의 속성들을 가지는 각각의 엔티티로 복수의 엔티티들을 나타내는 엔티티 데이터를 저장하도록 하고 - 상기 엔티티들 중 적어도 일부 각각은 여러 인스턴스들을 가지고, 상기 인스턴스들 중 적어도 일부 각각은 상기 속성들 중 적어도 하나에 대한 각각의 값들을 가지며, 상기 복수의 엔티티들은 복수의 입력 엔티티들과 복수의 출력 엔티티들을 포함함 - ;

입력 디바이스 또는 포트를 통해, 각각이 상기 입력 엔티티들 중 하나의 적어도 하나의 속성들과 상기 출력 엔티티들 중 하나의 적어도 하나의 출력 속성들 사이의 대응관계를 정의하는 적어도 하나의 매핑들을 포함하는 입력 데이터를 수신하도록 하고 - 상기 입력 엔티티는 상기 입력 엔티티에 대한 고유 키의 일부로서 식별된 적어

도 하나의 키 속성들을 포함하고, 상기 출력 엔티티는 상기 출력 엔티티에 대한 고유 키의 일부로서 식별된 적어도 하나의 키 속성들을 포함함 - ;

출력 디바이스 또는 포트를 통해, 상기 입력 데이터를 수신하도록 구성된 사용자 인터페이스를 디스플레이하도록 하고; 그리고

적어도 하나의 프로세서로, 상기 사용자 인터페이스에 디스플레이되는 결과 정보를 계산하도록 하는 - 상기 결과 정보는 상기 입력 데이터에 포함된 상기 적어도 하나의 매핑들에 따라 상기 출력 엔티티들의 인스턴스들을 생성하기 위해 상기 입력 엔티티들의 인스턴스들을 처리한 결과를 특성화함 - 명령들을 포함하되, 상기 계산은

제1 출력 엔티티의 인스턴스들을 생성하기 위해 제1 입력 엔티티의 인스턴스들을 처리하는 것;

상기 적어도 하나의 매핑들에 기반하여, 상기 제1 출력 엔티티의 적어도 하나의 키 속성들 각각에 대응하는 상기 제1 입력 엔티티의 적어도 하나의 매핑된 입력 속성들을 결정하는 것;

상기 결정된 적어도 하나의 매핑된 입력 속성들에 기반하여 상기 제1 출력 엔티티의 인스턴스들을 생성하는 것;

처리된 상기 제1 입력 엔티티의 인스턴스들의 총 수를 계산하는 것; 및

생성된 상기 제1 출력 엔티티의 인스턴스들의 총 수를 계산하는 것을 포함하는, 컴퓨터 판독 가능한 매체 상에 비 일시적 형태로 저장된 소프트웨어.

발명의 설명

기 술 분 야

[0001] 본 출원은 2014년 3월 14일에 출원된 미국 출원 번호 61/953,021에 대해 우선권을 주장한다.

[0002] 본 설명은 키드 엔티티들의 속성들을 매핑하는 것에 관한 것이다.

배 경 기 술

[0003] 다양한 시스템들이 입력(또는 "기점(origin)") 시스템 또는 포맷으로부터의 데이터를 출력 시스템(또는 "종점(destination)") 또는 포맷으로 매핑하는 능력을 가진다. 상기 매핑 프로세스는 매핑에 따라, 입력 데이터에 변환 함수를 적용하고 출력 데이터로 결과들을 저장하는 것을 포함할 수 있다. "매핑(mapping)"은 입력 데이터의 속성들과 출력 데이터의 속성들 사이의 관계들을 명시하는 것으로 정의될 수 있다. 상기 매핑 프로세스는 예를 들어, 출력 데이터로서 입력 데이터가 시스템으로 로딩되게 할 수 있거나, 입력 데이터가 출력 데이터로 변환되게 할 수 있거나 또는 둘 다일 수 있다. 상기 입력 또는 출력 데이터의 내용은 일부 경우에 다른 데이터의 특성들을 설명하는 메타데이터를 나타내는 데이터 값들을 포함할 수 있다. 일부 시스템에서, 매핑 동작들은 추출, 변환, 및 로드(ETL) 처리의 맥락으로 수행된다.

발명의 내용

해결하려는 과제

[0004] 본 발명의 목적은 키드 엔티티들의 속성들을 매핑하여 데이터를 처리하기 위한 컴퓨팅 시스템, 데이터 처리 방법, 및 컴퓨터 프로그램을 제공하기 위한 것이다.

과제의 해결 수단

[0005] 일 측면에서, 일반적으로, 컴퓨팅 시스템은 하나 이상의 속성들을 가지는 각각의 엔티티로 복수의 엔티티들을 나타내는 엔티티 데이터를 저장하는 데이터 저장 시스템 - 상기 엔티티들 중 적어도 일부 각각은 여러 인스턴스들을 가지고, 상기 인스턴스들 중 적어도 일부 각각은 상기 속성들 중 하나 이상에 대한 각각의 값들을 가지며, 상기 복수의 엔티티들은 복수의 입력 엔티티들과 복수의 출력 엔티티들을 포함함 - ; 각각이 상기 입력 엔티티들 중 하나의 하나 이상의 입력 속성들과 상기 출력 엔티티들 중 하나의 하나 이상의 출력 속성들 사이의 대응 관계를 정의하는 하나 이상의 매핑들을 포함하는 입력 데이터를 수신하기 위한 입력 디바이스 또는 포트 - 상기 입력 엔티티는 상기 입력 엔티티에 대한 고유 키의 일부로서 식별된 하나 이상의 키 속성들을 포함하고, 상기

출력 엔티티는 상기 출력 엔티티에 대한 고유 키의 일부로서 식별된 하나 이상의 키 속성들을 포함함 - ; 상기 입력 데이터를 수신하도록 구성된 사용자 인터페이스를 디스플레이하기 위한 출력 디바이스 또는 포트; 및 상기 사용자 인터페이스에 디스플레이되는 결과 정보를 계산하도록 구성된 적어도 하나의 프로세서 - 상기 결과 정보는 상기 입력 데이터에 포함된 상기 하나 이상의 매핑들에 따라 상기 출력 엔티티들의 인스턴스들을 생성하기 위해 상기 입력 엔티티들의 인스턴스들을 처리한 결과를 특성화함- 를 포함한다. 상기 계산은 제1 출력 엔티티의 인스턴스들을 생성하기 위해 제1 입력 엔티티의 인스턴스들을 처리하는 것; 상기 하나 이상의 매핑들에 기반하여, 상기 제1 출력 엔티티의 하나 이상의 키 속성들 각각에 대응하는 상기 제1 입력 엔티티의 하나 이상의 매핑된 입력 속성들을 결정하는 것; 상기 결정된 하나 이상의 매핑된 입력 속성들에 기반하여 상기 제1 출력 엔티티의 인스턴스들을 생성하는 것; 처리된 상기 제1 입력 엔티티의 인스턴스들의 총 수를 계산하는 것; 및 생성된 상기 제1 출력 엔티티의 인스턴스들의 총 수를 계산하는 것을 포함한다.

[0006] 상기 사용자 인터페이스에 상기 결과 정보를 디스플레이하는 것은 상기 제1 출력 엔티티의 표현과 관련하여 상기 제1 출력 엔티티의 인스턴스들의 총 수를 디스플레이하는 것을 포함한다.

[0007] 상기 사용자 인터페이스에 상기 결과 정보를 디스플레이하는 것은 상기 제1 입력 엔티티의 표현과 관련하여 상기 제1 입력 엔티티의 인스턴스들의 총 수를 디스플레이하는 것을 포함한다.

[0008] 상기 사용자 인터페이스에 상기 결과 정보를 디스플레이하는 것은 상기 디스플레이된 입력 엔티티의 속성들과 디스플레이된 출력 엔티티의 속성들 사이의 하나 이상의 매핑들을 나타내는 여러 요소들을 디스플레이하는 것, 상기 디스플레이된 입력 엔티티와 디스플레이된 출력 엔티티 사이의 임의의 매핑들에 대한 입력 데이터가 출력 속성을 (1) 동일한 이름의 입력 속성, 또는 (2) 상수 값으로 할당하는지 여부를 표시하는 각각의 요소에 대한 아이콘을 디스플레이하는 것을 포함한다.

[0009] 상기 출력 엔티티의 적어도 하나의 키 속성들 각각에 대응하는 상기 입력 엔티티의 적어도 하나의 매핑된 입력 속성들을 결정하는 것은 상기 하나 이상의 매핑된 입력 속성들이 상기 출력 엔티티의 각각의 키 속성들과 일대일 대응관계를 가지는지 여부를 결정하는 것을 포함한다.

[0010] 상기 계산은 상기 매핑된 입력 속성들이 (1) 상기 입력 엔티티의 키 속성들 모두, 또는 (2) 상기 입력 엔티티의 키 속성들 모두보다 적은 것을 포함하는지를 결정하기 위해 상기 매핑된 입력 속성들을 상기 입력 엔티티의 하나 이상의 키 속성들과 비교하는 것을 더 포함한다.

[0011] 상기 처리는 (1) 상기 매핑된 입력 속성들이 상기 입력 엔티티의 키 속성들 모두를 포함한다는 결정에 응하여, 상기 출력 엔티티의 인스턴스들과 매칭 키 속성들(matching key attribute)을 가지는 상기 입력 엔티티의 인스턴스들 사이의 일대일 대응관계, 또는 (2) 상기 매핑된 입력 속성들이 상기 입력 엔티티의 키 속성들 모두보다 적은 것을 포함한다는 결정에 응하여, 상기 매핑된 입력 속성들에 대해 동일한 값들을 공유하는 상기 입력 엔티티의 여러 인스턴스들의 집계(aggregation)에 기반하여 상기 출력 엔티티의 인스턴스들을 생성하는 것을 더 포함한다.

[0012] 상기 엔티티 데이터는 계층 구조에 따라 관련된 복수의 출력 엔티티들을 나타내고, 적어도 하나의 루트 출력 엔티티(root output entity)는 상기 계층 구조의 최상위 레벨에 있고 하나 이상의 출력 엔티티들은 상기 계층 구조의 최상위 레벨 아래의 하나 이상의 레벨들에 있으며, 상기 루트 엔티티보다 낮은 레벨의 각각의 출력 엔티티는 단일 출력 엔티티의 서브 엔티티(sub-entity)이다.

[0013] 상기 엔티티 데이터는 계층 구조에 따라 관련된 복수의 입력 엔티티들을 나타내고, 적어도 하나의 루트 입력 엔티티는 상기 계층 구조의 최상위 레벨에 있고 하나 이상의 입력 엔티티들이 상기 계층 구조의 최상위 레벨 아래의 하나 이상의 레벨들에 있으며, 상기 루트 엔티티보다 낮은 레벨의 각각의 입력 엔티티는 단일 입력 엔티티의 서브 엔티티이다.

[0014] 계층 구조에 따라 관련된 상기 복수의 출력 엔티티들에 관련되지 않은 적어도 제1 엔티티는 상기 입력 데이터에 포함된 상기 매핑들 중 적어도 하나에 의해 출력 속성으로 참조되는 적어도 하나의 속성을 포함한다.

[0015] 상기 제1 엔티티는 상기 입력 데이터에 포함된 상기 매핑들 중 적어도 하나에 의해 입력 속성으로 참조되는 적어도 하나의 속성을 포함한다.

[0016] 제2 엔티티의 서브 엔티티인 제1 엔티티의 복수의 인스턴스들 각각은 상기 제2 엔티티의 특정 인스턴스를 식별하는 상기 제1 엔티티의 키 속성의 공통 값을 포함한다.

[0017] 상기 제1 엔티티는 레코드들의 제1 세트에 대응하고, 상기 제2 엔티티는 레코드들의 제2 세트에 대응하며, 상기

제1 엔티티의 키 속성은 레코드들의 상기 제2 세트의 특정 레코드의 기본 키 필드에 포함된 값을 식별하는 레코드들의 상기 제1 세트의 외래 키 필드에 대응한다.

[0018] 제2 엔티티의 서브 엔티티인 제1 엔티티의 복수의 인스턴스들은 상기 제2 엔티티의 특정 인스턴스의 데이터 구조 내에 포함된 벡터(vector)의 복수의 요소들에 대응한다.

[0019] 상기 처리는 상기 출력 엔티티의 인스턴스들을 생성하기 위해 상기 입력 엔티티의 인스턴스들을 처리하도록 데이터플로 그래프를 이용하여 상기 출력 엔티티의 인스턴스들을 생성하는 것을 더 포함하고, 상기 데이터플로 그래프는 엔티티의 인스턴스들에 연산들을 수행하도록 구성된 컴포넌트들(components)을 나타내는 노드들과 컴포넌트들 사이의 인스턴스들의 흐름들을 나타내는 노드들 사이의 링크들을 포함한다.

[0020] 상기 데이터플로 그래프는 상기 하나 이상의 매핑들의 입력 속성들에 기반하여 또 다른 엔티티의 인스턴스의 데이터 구조로부터 서브 엔티티의 인스턴스들의 하나 이상의 벡터들을 추출하도록 구성되는 적어도 하나의 분할 컴포넌트(spilt component)와 상기 하나 이상의 매핑들의 출력 속성들에 기반하여 또 다른 엔티티의 인스턴스의 데이터 구조로 서브 엔티티의 인스턴스들의 하나 이상의 벡터들을 삽입하도록 구성된 적어도 하나의 결합 컴포넌트(combine component)를 포함한다.

[0021] 또 다른 측면에 있어서, 일반적으로, 컴퓨팅 시스템은 하나 이상의 속성들을 가지는 각각의 엔티티로 복수의 엔티티들을 나타내는 엔티티 데이터를 저장하기 위한 수단 - 상기 엔티티들 중 적어도 일부 각각은 여러 인스턴스들을 가지고, 상기 인스턴스들 중 적어도 일부 각각은 상기 속성들 중 하나 이상에 대한 각각의 값을 가지며, 상기 복수의 엔티티들은 복수의 입력 엔티티들과 복수의 출력 엔티티들을 포함함 - ; 각각이 상기 입력 엔티티들 중 하나의 하나 이상의 입력 속성들과 상기 출력 엔티티들 중 하나의 하나 이상의 출력 속성들 사이의 대응관계를 정의하는 하나 이상의 매핑들을 포함하는 입력 데이터를 수신하기 위한 수단 - 상기 입력 엔티티는 상기 입력 엔티티에 대한 고유 키의 일부로서 식별된 하나 이상의 키 속성들을 포함하고, 상기 출력 엔티티는 상기 출력 엔티티에 대한 고유 키의 일부로서 식별된 하나 이상의 키 속성들을 포함함 - ; 상기 입력 데이터를 수신하도록 구성된 사용자 인터페이스를 디스플레이하기 위한 수단; 및 상기 사용자 인터페이스에 디스플레이된 결과 정보를 계산하기 위한 수단 - 상기 결과 정보는 상기 입력 데이터에 포함된 상기 하나 이상의 매핑들에 따라 상기 출력 엔티티들의 인스턴스들을 생성하기 위해 상기 입력 엔티티들의 인스턴스들을 처리한 결과를 특성화함 - 을 포함한다. 상기 계산은 제1 출력 엔티티의 인스턴스들을 생성하기 위해 제1 입력 엔티티의 인스턴스들을 처리하는 것; 상기 하나 이상의 매핑들에 기반하여, 상기 제1 출력 엔티티의 하나 이상의 키 속성들 각각에 대응하는 상기 제1 입력 엔티티의 하나 이상의 매핑된 입력 속성들을 결정하는 것; 상기 결정된 하나 이상의 매핑된 입력 속성들에 기반하여 상기 제1 출력 엔티티의 인스턴스들을 생성하는 것; 처리된 상기 제1 입력 엔티티의 인스턴스들의 총 수를 계산하는 것; 및 생성된 상기 제1 출력 엔티티의 인스턴스들의 총 수를 계산하는 것을 포함한다.

[0022] 또 다른 측면에서, 일반적으로, 컴퓨팅 시스템에서 데이터를 처리하기 위한 방법은 데이터 저장 시스템에, 하나 이상의 속성들을 가지는 각각의 엔티티로 복수의 엔티티들을 나타내는 엔티티 데이터를 저장하는 단계 - 상기 엔티티들 중 적어도 일부 각각은 여러 인스턴스들을 가지고, 상기 인스턴스들 중 적어도 일부 각각은 상기 속성들 중 하나 이상에 대한 각각의 값들을 가지며, 상기 복수의 엔티티들은 복수의 입력 엔티티들과 복수의 출력 엔티티들을 포함함 - ; 입력 디바이스 또는 포트를 통해, 각각이 상기 입력 엔티티들 중 하나의 하나 이상의 입력 속성들과 상기 출력 엔티티들 중 하나의 하나 이상의 출력 속성들 사이의 대응관계를 정의하는 하나 이상의 매핑들을 포함하는 입력 데이터를 수신하는 단계 - 상기 입력 엔티티는 상기 입력 엔티티에 대한 고유 키의 일부로서 식별된 하나 이상의 키 속성들을 포함하고, 상기 출력 엔티티는 상기 출력 엔티티에 대한 고유 키의 일부로서 식별된 하나 이상의 키 속성들을 포함함 - 출력 디바이스 또는 포트를 통해, 상기 입력 데이터를 수신하도록 구성된 사용자 인터페이스를 디스플레이하는 단계; 및 적어도 하나의 프로세서로, 상기 사용자 인터페이스에 디스플레이되는 결과 정보를 계산하는 단계 - 상기 결과 정보는 상기 입력 데이터에 포함된 상기 하나 이상의 매핑들에 따라 상기 출력 엔티티들의 인스턴스들을 생성하기 위해 상기 입력 엔티티들의 인스턴스들을 처리한 결과를 특성화함 - 를 포함한다. 상기 계산하는 단계는 제1 출력 엔티티의 인스턴스들을 생성하기 위해 제1 입력 엔티티의 인스턴스들을 처리하는 단계; 상기 하나 이상의 매핑들에 기반하여, 상기 제1 출력 엔티티의 하나 이상의 키 속성들 각각에 대응하는 상기 제1 입력 엔티티의 하나 이상의 매핑된 입력 속성들을 결정하는 단계; 상기 결정된 하나 이상의 매핑된 입력 속성들에 기반하여 상기 제1 출력 엔티티의 인스턴스들을 생성하는 단계; 처리된 상기 제1 입력 엔티티의 인스턴스들의 총 수를 계산하는 단계; 및 생성된 상기 제1 출력 엔티티의 인스턴스들의 총 수를 계산하는 단계를 포함한다.

[0023] 또 다른 측면에 있어서, 일반적으로, 소프트웨어가 컴퓨터 판독 가능한 매체 상에 비 일시적 형태로 저장되고, 상기 소프트웨어는 컴퓨팅 시스템이 데이터 저장 시스템에, 하나 이상의 속성들을 가지는 각각의 엔티티로 복수의 엔티티들을 나타내는 엔티티 데이터를 저장하도록 하고 - 상기 엔티티들 중 적어도 일부 각각은 여러 인스턴스들을 가지고, 상기 인스턴스들 중 적어도 일부 각각은 상기 속성들 중 하나 이상에 대한 각각의 값들을 가지며, 상기 복수의 엔티티들은 복수의 입력 엔티티들과 복수의 출력 엔티티들을 포함함 - ; 입력 디바이스 또는 포트를 통해, 각각이 상기 입력 엔티티들 중 하나의 하나 이상의 속성들과 상기 출력 엔티티들 중 하나의 하나 이상의 출력 속성들 사이의 대응관계를 정의하는 하나 이상의 매핑들을 포함하는 입력 데이터를 수신하도록 하고 - 상기 입력 엔티티는 상기 입력 엔티티에 대한 고유 키의 일부로서 식별된 하나 이상의 키 속성들을 포함하고, 상기 출력 엔티티는 상기 출력 엔티티에 대한 고유 키의 일부로서 식별된 하나 이상의 키 속성들을 포함함 - ; 출력 디바이스 또는 포트를 통해, 상기 입력 데이터를 수신하도록 구성된 사용자 인터페이스를 디스플레이하도록 하고; 그리고 적어도 하나의 프로세서로, 상기 사용자 인터페이스에 디스플레이되는 결과 정보를 계산하도록 하는 - 상기 결과 정보는 상기 입력 데이터에 포함된 상기 하나 이상의 매핑들에 따라 상기 출력 엔티티들의 인스턴스들을 생성하기 위해 상기 입력 엔티티들의 인스턴스들을 처리한 결과를 특성화함 - 명령들을 포함한다. 상기 계산은 제1 출력 엔티티의 인스턴스들을 생성하기 위해 제1 입력 엔티티의 인스턴스들을 처리하는 것; 상기 하나 이상의 매핑들에 기반하여, 상기 제1 출력 엔티티의 하나 이상의 키 속성들 각각에 대응하는 상기 제1 입력 엔티티의 하나 이상의 매핑된 입력 속성들을 결정하는 것; 상기 결정된 하나 이상의 매핑된 입력 속성들에 기반하여 상기 제1 출력 엔티티의 인스턴스들을 생성하는 것; 처리된 상기 제1 입력 엔티티의 인스턴스들의 총 수를 계산하는 것; 및 생성된 상기 제1 출력 엔티티의 인스턴스들의 총 수를 계산하는 것을 포함한다.

발명의 효과

[0024] 측면들은 다음 장점들 중 하나 이상을 포함할 수 있다.

[0025] 상기 매핑 기술들은 입력 데이터와 출력 데이터 내에 존재하는 특정 엔티티들의 고유 인스턴스들을 식별하기 위한 특정 특성들을 보존하는 한편, 입력 데이터를 출력 데이터로 매핑하는데 유연성이 있게 한다. 상기 입력 또는 출력 데이터는 하나 이상의 엔티티들을 나타내는 "엔티티 데이터"를 포함할 수 있다. 엔티티는 독립적 존재가 가능하거나 고유하게 식별될 수 있는, 정보 도메인에서, 특정 종류의 임의의 수의 아이템들의 컬렉션의 추상화(abstraction)로 간주될 수 있다. 예를 들어, "계좌(Accounts)" 엔티티는 데이터베이스의 테이블에 의해, 또는 파일(예를 들어, 구분된 레코드들(delimited records)을 가짐)로 저장된 데이터세트 에 의해 나타내어질 수 있다. 데이터베이스 테이블 또는 데이터세트 파일의 개별 레코드들(또는 "행들(rows)")은 각각 예를 들어, 금융 또는 상업 데이터를 관리하는 시스템에서 특정 계좌 소유자에 대한 계좌 엔티티의 상이한 인스턴스를 나타낼 수 있다. 엔티티는 또한 특정 클래스의 데이터 객체들의 컬렉션 같은 임의의 다른 유형의 데이터 구조에 의해 나타내어질 수 있고, 여기서 상기 엔티티의 상이한 인스턴스들이 상기 데이터 객체의 상이한 인스턴스들에 대응한다. 각각의 엔티티는 임의의 수의 속성들을 가질 수 있다. 예를 들어, 데이터베이스 테이블에 의해 나타내어지는 엔티티에서, 상기 테이블의 필드들(또는 "열들(columns)")은 그 엔티티의 특정 속성에 대응하는 특정 유형의 데이터(예를 들어, 소정의 데이터 유형을 가지는 변수)를 저장하기 위해 정의될 수 있다. 계좌 엔티티에 대한 테이블은 예를 들어, “first_name," "last_name", 및 (사회 보장 번호에 대한)"SSN" 이라고 라벨 붙여진 필드들을 포함할 수 있고, 상기 테이블의 레코드들(계좌 엔티티의 인스턴스들을 나타냄)은 각각 상기 필드들 각각에 대한 각각의 값들을 가질 수 있다.

[0026] 엔티티의 상이한 인스턴스들이 고유하게 식별될 수 있는지를 확인하기 위해, 상기 엔티티의 하나 이상의 속성들은 상기 엔티티에 대한 고유 키의 일부인 "키 속성들(key attributes)"로 식별된다. 일부 경우에, 엔티티는 단일 키 속성을 가진다. 예를 들어, “master_account_number"라고 라벨 붙여진 필드는 계좌 엔티티의 인스턴스를 나타내는 각각의 계좌 레코드에 대해 고유한 값을 저장할 수 있다. 이러한 단일 키 필드는 때때로 "단순 키(simple key)"라고 한다. 일부 경우에, 엔티티는 고유 키를 함께 형성하는 다중 키 속성들("복합 키(compound key)"라고도 함)을 가진다. 예를 들어, “first_name," "last_name", 및 "SSN" 필드들의 조합(예를 들어, 연결(concatenation))은 함께 계좌 엔티티의 인스턴스를 나타내는 레코드를 고유하게 식별하는 키 속성들로서 작용할 수 있다. 고유 값들("후보 키들(candidate keys)"이라고도 함)을 가지는 여러 필드들이 있을 수 있고, 그 필드들 중 하나(또는 필드들의 조합)가 사용될 고유 키("기본 키(primary key)"라고도 함)로서 사용을 위해 선택될 수 있다. 때때로, 필드가 고유 키의 일부로서 작용할 값("대체 키(surrogate key)"라고도 함)을 저장하기 위해 레코드에 부가된다.

[0027] 데이터 처리 시스템에서 특정 데이터를 처리하는 것을 시도하는 사용자에 대해 발생할 수 있는 문제는 상기 처

리가 키 속성들로서 특정 필드를 필요로 할 수 있으나, 기존 데이터가 키 속성들로서 다른 필드들을 가질 수 있다는 것이다. 그러나, 키 필드들은 데이터가 실제로 정확한 특성들(즉, 키의 각각의 고유 값에 대해 단일 레코드가 있음)을 가지는지 확인하지 않고는 변경될 수 없다. 이러한 재구성은 수천 또는 수백만 레코드들이 있을 수 있는 현실적인 산업 응용에서 사용자가 수행하는 데 실용적이지 않을 수 있다. 본원에서 설명된 상기 기술들은 사용자가 입력 데이터를 레코드 단위로 재구성(또는 처음부터 그렇게 할 프로그램을 작성)할 것을 요구하지 않고 키 변경이 요구될 때조차 처리가 효율적으로 수행될 수 있게 한다. 예를 들어, 상기 기술들은 특정 환경에서 필요로 될 수 있는 임의의 집계(aggregation)(예를 들어, 특정 키 값에 대한 여러 레코드들로부터의 데이터를 집계하는 것)가 키 속성들로 원하는 필드들을 사용하여 적용될 것을 보장한다. 레코드들을 식별하는 입력 데이터 처리 결과는 그런 다음 입력과 출력 레코드들의 총 수의 형태로 사용자 인터페이스 내에 디스플레이될 수 있다.

[0028] 특정 엔티티와 그 속성들을 나타내는 엔티티 데이터의 구조는 레코드 내 필드들을 정의하는 데이터베이스 테이블 또는 데이터세트 파일에 대한 레코드 포맷같은 포맷 정보에 의해 정의될 수 있다. 각각의 필드에 나타나는 값들의 데이터 유형들과 바이트 길이들에 부가하여, 레코드 포맷은 어떤 필드들이 기본 키를 구성하는 키 필드들로서 사용될 것인지를 정의할 수 있다. 매핑 절차들은 사용자가 출력 엔티티의 어떤 속성들이 키 속성들이 될 것인지를 정의할 수 있게 한다. 그 출력 키 속성들 중 일부는 입력 키 속성들에 매핑되었을 수 있거나, 그 출력 키 속성들 중 일부는 입력 엔티티의 비-키 속성들(non-key attributes)에 매핑되었을 수 있다. 그 출력 키 속성들에 매핑되었던 입력 속성들을 입력 키 속성들과 자동적으로 비교하여, 상기 시스템은 출력 엔티티들의 인스턴스들을 고유하게 식별할 수 있는 잘 정의된 키 속성들을 유지하는 방법으로 매핑에 따라 출력 엔티티의 인스턴스들을 생성하는 방법을 결정할 수 있다. 입력 데이터에 의해 나타내어지는 입력 엔티티들을 출력 데이터에 의해 나타내어지는 출력 엔티티로의 매핑은 매핑된 출력 데이터가 입력 데이터보다 더 효율적으로 처리되고/되거나 관리될 수 있게 할 수 있다. 일부 경우에, 여러 관련된 엔티티들에 대한 엔티티 데이터는 하기에 더 상세히 설명된 바와 같이 상기 엔티티들의 인스턴스들 간의 계층적 관계를 정의할 수 있다. 상기 매핑 절차들은 이러한 계층 구조들을 재구성할 수 있고 상기 엔티티들이 잘 정의된 키 속성들을 여전히 유지하는 것을 보장할 수 있다.

[0029] 본 발명의 다른 특징들 및 장점들은 다음 설명으로부터, 그리고 청구범위로부터 명백해질 것이다.

도면의 간단한 설명

[0030] 도 1은 데이터 처리 시스템의 블록도이다.

도 2a-2b는 엔티티-관계 다이어그램들이다.

도 3a-3d는 사용자 인터페이스의 부분들의 예의 스크린샷들(screenshots)이다.

도 4는 데이터플로 그래프들(dataflow graphs)을 생성하기 위한 절차의 순서도이다.

도 5는 데이터플로 그래프의 다이어그램이다.

발명을 실시하기 위한 구체적인 내용

[0031] 도 1a는 매핑 기술들이 사용될 수 있는 데이터 처리 시스템(100)의 예를 도시한다. 상기 시스템(100)은 저장 디바이스들 또는 온라인 데이터 스트림들에 대한 연결들 - 그것들 각각이 다양한 포맷들 중 임의의 것(예를 들어, 데이터베이스 테이블들, 스프레드시트 파일들, 플랫 텍스트 파일들(flat text files), 또는 메인프레임에 의해 사용되는 원시 포맷)으로 데이터를 저장할 수 있거나 제공할 수 있음 - 같은 데이터의 하나 이상의 소스들을 포함할 수 있는 데이터 관리 시스템(102)을 포함한다. 실행 환경(104)은 매핑 모듈(mapping module)(106)과 실행 모듈(112)을 포함한다. 상기 실행환경(104)은 예를 들어, UNIX 운영 시스템의 버전 같은 적절한 운영 시스템의 제어하의 하나 이상의 범용 컴퓨터들에 호스팅될 수 있다. 예를 들어, 상기 실행 환경(104)은 지역(예를 들어, 대칭적 멀티 프로세싱(SMP) 컴퓨터들 같은 멀티프로세서 시스템들) 또는 지역적으로 분산된(예를 들어, 클러스터들 또는 대량 병렬 처리(MPPs) 시스템들처럼 결합된 여러 프로세서들), 또는 원격, 또는 원격적으로 분산된(예를 들어, 지역 네트워크(LAN) 및/또는 광역 네트워크(WAN)를 통해 결합된 여러 프로세서들) 또는 그 임의의 조합인 여러 중앙 처리 유닛들(CPUs) 또는 프로세서 코어들을 사용하는 컴퓨터 시스템들의 구성을 포함하는 다중-노드 병렬 컴퓨팅 환경을 포함할 수 있다.

[0032] 상기 매핑 모듈(106)은 상기 실행 환경(104)에 액세스 가능한 데이터 저장 시스템(116)에 저장된 하나 이상의

매핑들(114)에 기반하여 데이터 관리 시스템(102)로부터 입력 데이터를 판독하고 입력 데이터의 엔티티들을 출력 데이터의 엔티티들로 매핑하도록 구성된다. 상기 매핑들(114)은 각각 입력 엔티티의 하나 이상의 입력 속성들과 출력 엔티티의 하나 이상의 출력 속성들 사이의 대응관계(correspondence)를 정의한다. 예를 들어, 상기 대응관계는 두 속성들 사이의 등식(equality) 또는 하나의 속성을 또 다른 속성의 함수로서 정의하는 표현식(expression)일 수 있다. 상기 출력 데이터는 상기 데이터 관리 시스템(102)에 또는 데이터 저장 시스템(116)에 다시 저장될 수 있거나, 그렇지 않으면 사용될 수 있다. 상기 데이터 저장 시스템(116)은 임의 레벨의 캐시 메모리, 또는 동적 랜덤 액세스 메모리(DRAM)의 메인 메모리같은 휘발성 저장 매체, 또는 자기 하드 디스크 드라이브(들)같은 비휘발성 저장을 포함하여 저장 매체의 임의의 조합을 포함할 수 있다. 상기 데이터 관리 시스템(102)을 제공하는 저장 디바이스들은 예를 들어, 상기 실행 환경(104)을 호스팅하는 컴퓨터에 연결된 저장 매체(예를 들어, 하드 드라이브(108))상에 저장되는 실행 환경(104)에 로컬일 수 있거나, 예를 들어, 원격 연결을 통해(예를 들어, 클라우드 컴퓨팅 인프라스트럭처에 의해 제공됨) 상기 실행 환경(104)을 호스팅하는 컴퓨터와 통신하는 원격 시스템(예를 들어, 메인프레임(110))상에 호스팅되는 실행 환경(104)에 원격일 수 있다.

[0033] 상기 실행 모듈(112)은 데이터 처리 작업들을 수행하기 위해 상기 매핑 모듈(106)에 의해 생성되는 출력 데이터를 사용하고, 그것들 중 일부는 상기 매핑들(114)에 의해 정의되었던 출력 데이터의 데이터 포맷에 의존할 수 있다. 상기 시스템(100)은 또한 사용자(120)가 상기 매핑들(114), 및 상기 실행 모듈(112)에 의해 실행되는 데이터 처리 프로그램의 다른 측면들을 정의할 수 있는 사용자 인터페이스(118)(예를 들어, 상기 실행 환경(104)과 통신하거나 상기 실행 환경을 호스팅하는 컴퓨터의 디스플레이의 스크린 상에 디스플레이되는 그래픽 사용자 인터페이스)를 포함한다. 상기 시스템(100)은 일부 구현에서, 정점들 사이에 방향성 링크들(directed links)(작업 요소들, 즉 데이터의 흐름을 나타냄)에 의해 연결된 정점들(데이터 처리 컴포넌트들 또는 데이터세트들을 나타냄)을 포함하는 데이터플로 그래프들(dataflow graphs)과 같은 어플리케이션들을 개발하기 위해 구성된다. 예를 들어, 이러한 환경은 본원에서 참조로 원용된"그래프기반 어플리케이션들에 대한 파라미터들 관리(Managing Parameters for Graph-Based Applications)"라는 타이틀의 미국 공보 번호 2007/0011668에서 더 상세히 설명된다. 이러한 그래프 기반 계산들을 실행하기 위한 시스템이 본원에서 참조로 원용된"그래프들로 표현된 계산의 실행 (EXECUTING COMPUTATIONS EXPRESSED AS GRAPHS)"이라는 타이틀의 미국 특허 5,966,072에서 설명된다. 이 시스템에 따라 만들어진 데이터플로 그래프들은 프로세스들 사이에 정보를 이동시키기 위해 그리고 상기 프로세스들에 대한 실행 순서를 정의하기 위해 그래프 컴포넌트들에 의해 나타내어진 개별 프로세스들로 정보를 보내고 개별 프로세스들로부터 정보를 얻기 위한 방법들을 제공한다. 이 시스템은 임의의 가용한 방법들(예를 들어, 그래프의 링크들에 따른 통신 경로들은 TCP/IP 또는 UNIX 도메인 소켓들을 사용할 수 있거나, 프로세스들 사이에서 데이터를 전달하기 위해 공유 메모리를 사용할 수 있음)로부터 프로세스간 통신 방법들을 선택하는 알고리즘들을 포함한다.

[0034] 상기 매핑 모듈(106)은 예를 들어, 데이터 세트 파일들 또는 데이터베이스 테이블들을 포함하여 상기 데이터 관리 시스템(102)으로부터 액세스가능한 입력 데이터 내에서 나타내어질 수 있는 다양한 유형의 엔티티들의 속성들을 매핑할 수 있다. 상기 엔티티의 데이터 콘텐츠는 아마도 널(null) 값들을 포함하여 각각의 속성들("필드들" 또는 "속성들"이라고도 함)에 대한 값들을 가지는 레코드들로 구성될 수 있다. 상기 매핑 모듈(106)은 일반적으로 그 엔티티의 레코드들에 관한 어떤 초기 포맷 정보로 시작한다. 어떤 경우에는, 입력 데이터의 엔티티들의 레코드 구조는 초기에 공지되지 않을 수 있고 대신 입력 데이터의 분석 후에 결정될 수 있다. 레코드들에 관한 초기 정보는 예를 들어, 별개 값(distinct value)을 나타내는 비트들의 수, 레코드 내의 필드들의 순서, 및 상기 비트들에 의해 나타내어지는 값의 유형(예를 들어, 문자열, 부호있는/부호없는 정수)을 포함할 수 있다.

[0035] 일부 입력 데이터 또는 출력 데이터에 대해, 상기 엔티티들은 계층 구조를 가질 수 있고, 상기 엔티티들이 계층 구조에 따라 서로 관련된다. 일반적으로, 상기 계층 구조는 방향성 에지들(directed edges)에 의해 연결되는 정점들의 그래프(예를 들어, 방향성 비순환 그래프(DAG))로서 나타내어질 수 있고, 여기서 상기 정점들은 엔티티들을 나타내고, 상기 에지들은 상기 엔티티들 간의 관계들(relationships)을 나타낸다. 일부 구현에서, 상기 관계는 상기 엔티티들 간의 기본 키/외래 키 관계에 대응한다. 다른 구현에서, 상기 관계는 또 다른 엔티티의 인스턴스의 속성 내에 한 엔티티의 인스턴스의 네스팅(nesting)에 대응한다. 각각의 정점은 상기 계층 구조의 특정 레벨에 있다. 적어도 하나의 엔티티(예를 들어, 상기 계층 구조가 트리 구조를 가진다면 루트 엔티티)는 상기 계층 구조의 최상위 레벨에 있고, 하나 이상의 엔티티들이 상기 계층 구조의 최상위 레벨 아래 하나 이상의 레벨들에 있다. 최상위 레벨보다 낮은 레벨의 각각의 엔티티는 단일 상위 레벨 엔티티(또는 "부모 엔티티")의 서브-엔티티(또는 "자식 엔티티")이다. 예를 들어, 상기 관계들이 기본 키/외래 키 관계들일 때, 자식 엔티티의 인스턴스는 그 값이 부모 엔티티의 특정 인스턴스의 고유한 기본 키 값인 외래 키 필드를 가진다. 상기 관계들

이 네스팅 관계들(nesting relationships)일 때, 자식 엔티티의 인스턴스는 (예를 들어, 자식 인스턴스 데이터 구조 자체 또는 부모 인스턴스의 속성 내에 자식 인스턴스 데이터 구조에 대한 포인터를 저장하여) 부모 엔티티의 특정 인스턴스의 속성 내에 포함된다.

[0036] 이러한 계층 구조는 엔티티-관계(ER) 다이어그램에 그래픽으로 나타내어질 수 있다. 도 2a는 트리 구조를 가지는 엔티티들의 입력 계층 구조(200)의 예에 대한 ER 다이어그램을 도시한다. 최상위 레벨에서, "Accounts" 엔티티는 속성 라벨 뒤에 "(K)"에 의해 표시되는 바와 같이 키 속성인 “master_account_number"라고 라벨 붙여진 단일 속성을 가진다. Accounts 엔티티에 대한 다른 키 속성들이 없기 때문에, master_account_number의 값이 Accounts 엔티티의 상이한 인스턴스들을 고유하게 식별한다. 상기 Accounts 엔티티는 또한 두 자식 엔티티들("CheckingAccounts" 엔티티, 및 "SavingsAccounts" 엔티티)에 대한 관계들에 대한 속성들을 가진다. 부모 엔티티와 각각의 자식 엔티티 사이의 다이어그램(200)의 연결자들(connectors)은 일대다 관계를 표시하고, 그것은 부모 엔티티의 하나의 인스턴스에 대해, 자식 엔티티의 0, 하나, 또는 많은 관련된 인스턴스들이 있다는 것을 의미한다. 이러한 일대다 관계는 자식 엔티티에서 크로우즈 풋(crow's foot)으로 끝나는 부모 엔티티와 자식 엔티티 사이의 선으로 도시된다.

[0037] 상기 CheckingAccounts 엔티티는 두 개의 키 속성들("master_account_number"라고 라벨 붙여진 속성 및 "acct_id"라고 라벨 붙여진 속성)을 가진다. 상기 master_account_number 속성은 외래 키이고, 그것은 부모 Accounts 엔티티의 관련된 인스턴스의 기본 키의 특정 값을 저장한다. 상기 acct_id 속성은 서로 상이한 당좌 예금 계좌들(checking accounts)을, 그것들이 Accounts 엔티티의 동일한 마스터 계좌 인스턴스(master account instance)의 자식들일지라도(예를 들어, 특정 마스터 계좌와 관련된 계좌 소유자가 여러 당좌 예금 계좌들을 가진다면), 고유하게 구별하는 복합 키를 형성하는 부가적인 키 속성이다. 유사하게, SavingsAccounts 엔티티는 두 개의 키 속성들("master_account_number" 라고 라벨 붙여진 속성 및 "acct_id" 라고 라벨 붙여진 속성)을 가지고, 그것들은 또한 임의의 수의 예금 계좌들이 서로 고유하게 구별하는 것을 가능하게 한다. CheckingAccounts와 SavingsAccounts 엔티티들 각각은 또한 이러한 엔티티들에 대한 비 키 속성들인 다른 속성들("first_name", "last_name", "SSN", "balance" 및 “interest_rate")을 가진다.

[0038] 도 2b는 엔티티들의 출력 계층 구조(210)의 일 예에 대한 ER 다이어그램을 도시하고, 그것은 또한 트리 구조를 가지나 입력 계층 구조(200)와는 상이한 수의 엔티티들을 가진다. 상기 매핑 모듈(106)은 (예를 들어, 사용자로부터) 출력 계층 구조(210)의 일부로서 생성되는 “AccountHolders" 출력 엔티티를 명시하는 매핑을 수신했다. 이 예에서, 출력 계층 구조(210)의 일부인 나머지 출력 엔티티들(즉, 최상위 엔티티 Accounts, 그리고 그 자식 엔티티들 CheckingAccounts와 SavingsAccounts)는 입력 계층 구조(200)에서 발견되는 대응하는 라벨이 붙여진 엔티티들로부터 매핑된다. 상기 AccountHolders 엔티티는 CheckingAccounts 엔티티의 하나 이상의 인스턴스들 및/또는 SavingsAccounts 엔티티의 하나 이상의 인스턴스들로부터 도출된 각각의 계좌 소유자에 대한 속성들을 가지는 인스턴스들을 가진다. 특히, AccountHolders 엔티티의 인스턴스의 속성들 중 네 개(“master_account_number", "first_name", "last_name", 및 "SSN")는 CheckingAccounts 또는 SavingsAccounts 엔티티들 중 하나의 인스턴스의 대응하는 라벨이 붙여진 속성들로부터 도출되고, AccountHolders 엔티티의 인스턴스의 속성들 중 하나("balance")가 하기에 더 상세히 설명된 바와 같이 여러 인스턴스들에 대해 집계 함수(aggregation function)에 기반하여 계산된다. AccountHolders 엔티티는 두 개의 키 속성들(master_account_number 및 SSN)을 가진다. 상기 master_account_number 속성은 여전히 외래 키이고, 그것은 부모 Accounts 엔티티의 관련된 인스턴스의 기본 키의 특정 값을 저장한다. 상기 SSN 속성(계좌 소유자의 사회보장 번호를 저장함)은 서로 상이한 계좌 소유자들(즉, AccountHolders 엔티티의 인스턴스들)을, 그것들이 Accounts 엔티티의 동일한 마스터 계좌 인스턴스의 자식들일지라도, 고유하게 구별하는 복합 키를 형성하는 부가적인 키 속성이다.

[0039] 도 3a는 입력 섹션(302A)에 디스플레이된 입력 계층 구조에 대한 측면에서 출력 섹션(302B)에 디스플레이된 출력 계층 구조를 정의하기 위한 사용자 인터페이스(300)의 일 예의 스크린 샷을 도시한다. 상기 스크린 샷에 도시된 상기 사용자 인터페이스(300)의 상태는 사용자가 소스에서 타깃으로의 매핑 섹션(Source-to-Target mappings section)(304)내에서 원하는 매핑들(114)을 정의하는 정보를 공급했고 입력 계층 구조의 레코드들로부터 출력 계층 구조의 레코드들을 생성하는 변환을 실행한 일 예에 해당한다. 상기 입력 계층 구조는 상기 시스템(100)에 의해 해석될 수 있는 구문(예를 들어, 데이터 조작 언어(DML) 구문, 또는 확장가능 마크업 언어(XML) 구문)측면에서 정의된 레코드 포맷 같은 저장된 포맷 정보, 또는 데이터베이스 테이블 스키마(schema)에 따라 디스플레이된다. 다음은 입력 레코드의 필드들로서 입력 속성들을 정의하는 DML 구문을 사용하는 이 예에서 입력 계층 구조를 명시하는 레코드 포맷의 일 예이다.

```
record
 decimal(",") master_account_number;
 record
  string(",") first_name;
  string(",") last_name;
  string(",") SSN;
  string(",") acct_id;
  decimal(",") balance;
  decimal(",") interest_rate;
 end[decimal(4)] checking_accounts;
 record
  string(",") first_name;
  string(",") last_name;
  string(",") SSN;
  string(",") acct_id;
  decimal(",") balance;
  decimal(",") interest_rate;
 end[decimal(4)] savings_accounts;
 string("\n") new_line= "\n";
end;
```

[0040]

[0041] "record"와 "end"키워드들의 외부 쌍은 최상위("in")엔티티를 나타내는 레코드를 정의한다. "record"와 "end"키워드들의 내부 쌍은 자식(checking_accounts와 savings_accounts) 엔티티들을 나타내는 레코드들을 정의한다. 상기 엔티티들의 속성들을 나타내는 필드들은 "record"와 "end" 키워드들 사이에 열거된다. 레코드 포맷은 반드시 그 레코드에 의해 나타내어지는 고 레벨 엔티티의 일부인 것은 아닌 값들을 저장하기 위해 레코드들에 포함되는 필드들을 정의할 수 있다. 이 예에서, new_line 필드는 레코드 포맷의 checking_accounts와 savings_accounts 레코드들 뒤에 나타나고, "in"엔티티의 속성으로 사용되는게 아니라, 오히려 예를 들어, 텍스트 에디터(text editor)에 디스플레이되는 목록에서 "in"엔티티의 인스턴스들을 나타내는 상이한 실제 레코드들 사이에 하드 코딩된 새로운 라인 문자(hard coded new line character)를 제공하는 구문 요소로 사용된다.

[0042] 상기 매핑 모듈(106)은 소스에서 타깃으로의 매핑 섹션(304) 내에서 정의된 매핑들(114)에 따라 "out"엔티티의 인스턴스들을 나타내는 레코드들에 대해 사용되는 적절한 레코드 포맷을 생성한다. 다음은 이 예에서 출력 레코드의 필드들로 출력 속성들을 정의하는 동일한 DML 구문을 사영하여 출력 계층 구조를 명시하는 레코드 포맷의 일 예이다.

```
record
 decimal(",") master_account_number;
 record
  string(",") first_name;
  string(",") last_name;
  string(",") SSN;
  decimal(",") balance;
 end[decimal(4)] account_holders;
 record
  string(",") acct_id;
  string(",") SSN;
  decimal(",") balance;
  decimal(",") interest_rate;
 end[decimal(4)] checking_accounts;
 record
  string(",") acct_id;
  string(",") SSN;
  decimal(",") balance;
  decimal(",") interest_rate;
 end[decimal(4)] savings_accounts;
 string("\n") new_line= "\n";
end;
```

[0043]

[0044] 이 출력 레코드 포맷은 사용자가 출력 계층 구조에서 다양한 엔티티들의 속성들에 대한 매핑들을 제공한 후에 생성되고, 사용자는 각각의 출력 엔티티의 속성들 중 어떤 것이 키 속성들로 사용될 것인지를 (예를 들어, 출력 섹션(302B)내에서) 식별할 수 있다. 출력 엔티티들의 어떤 속성들이 키 속성들인지에 관한 이러한 정보, 및 입력 엔티티들의 어떤 속성들이 그 키 속성들(출력 키의 "역 이미지(inverse image)"라고 함)에 매핑되었는지에 관한 정보가 데이터플로 그래프를 생성하는 데 사용되고, 그런 다음 그것은 하기에 더 상세히 설명된 바와 같이 출력 계층 구조의 엔티티들의 인스턴스들을 나타내는 실제 레코드들을 생성하기 위해 실행된다.

[0045] 디스플레이된 사용자 인터페이스(300)는 "in"이라 라벨 붙여진 입력 섹션 (302A)의 상단의 입력 계층 구조의 최상위 엔티티를 나타내는 아이콘(테이블을 묘사함), 그리고 "out"이라고 라벨 붙여진 출력 섹션(302B)의 상단의 출력 계층 구조의 최상위 엔티티를 나타내는 아이콘(테이블을 묘사함)을 포함한다. 각각의 엔티티의 인스턴스들의 수가 대괄호 안에서 라벨 옆에 디스플레이된다. 예를 들어, 출력 계층 구조의 레코드들이 생성된 후에, "[5 recs]"가 그 엔티티의 상이한 각각의 인스턴스들의 내용을 저장하는 5개의 레코드들이 있음을 표시하며 최상위 엔티티들 둘 모두에 대해 디스플레이된다. 이 예에서, 최상위 입력 엔티티와 출력 엔티티는 각각 도 2a와 2b의 ER 다이어그램의 Accounts 엔티티들에 해당한다. 이러한 최상위 엔티티들 각각은 최상위 엔티티에 대한 아이콘 바로 아래에 나타나는 아이콘 뒤에 디스플레이되는 키 속성 master_account_number를 나타내는 필드를 포함하여 ER 다이어그램에 도시된 바와 동일한 속성들과 서브- 엔티티들을 나타내는 필드들을 포함한다. 속성들에 대응하는 필드들이 "문자열(string)" 유형을 가지는 값으로 레코드들에 나타나는 것을 표시하는 글자 "A"를 묘사하는 아이콘들과 함께, 또는 "십진수(decimal)" 유형을 가지는 값으로 레코드들에서 나타나는 것을 표시하는 숫자

"12"를 묘사하는 아이콘들과 함께 디스플레이된다. 사용자 인터페이스(300)에서, 키의 일부인 각각의 필드(즉, 키 속성)가 필드의 아이콘 다음에 나타나는 키를 묘사하는 아이콘에 의해 사용자 인터페이스(300)내에서 식별된다.

[0046] 사용자 인터페이스(300)는 입력 섹션(302A)과 출력 섹션(302B)이 옵션 섹션(306A)와 옵션 섹션(306B)에서 각각 선택가능한 상이한 보기 모드들에서 보여지는 것이 가능하게 한다. "계층 구조 보기 모드(hierarchy view mode)"에서, 부모 엔티티의 서브-엔티티들에 대한 테이블 아이콘들이 그 부모 엔티티의 속성들과 동일한 양으로 들여쓰기 되어 디스플레이되고, 부모 엔티티의 키 속성을 참조하는 키 속성들은 자식 엔티티에서 도시되지 않는다. 도 3a는 계층 구조 보기 모드의 입력 섹션(302A)와 출력 섹션(302B) 둘 모두를 보여준다. 입력 섹션(302A)에서, checking_accounts 엔티티와 savings_accounts 엔티티에 대한 테이블 아이콘들이 master_account_number 키 속성에 대한 아이콘 아래에 그리고 그것과 수평으로 정렬되어 나타난다. 출력 섹션(302B)에 대해, account_holders 엔티티 그리고 checking_accounts 엔티티 그리고 savings_accounts 엔티티에 대한 테이블 아이콘들이 master_account_number 키 속성에 대한 아이콘 아래에 그리고 그것과 수평으로 정렬되어 나타난다.

[0047] 적어도 하나의 서브-엔티티를 가지는 각각의 엔티티는 하나 이상의 키 속성들로 구성된 키를 가진다. 이는 각각의 서브-엔티티가 서브-엔티티의 각각의 인스턴스에 대해 그 서브-엔티티에 관련된 부모 엔티티의 고유 인스턴스를 식별하는 대응하는 외래-키 속성을 가지게 할 수 있다. 부모 엔티티의 키의 (외래 키) 값을 저장하는 키 속성의 존재는 이러한 속성들을 디스플레이하지 않는 계층 구조 보기 모드에서 암시된다. 예를 들어, 입력 계층 구조와 출력 계층 구조 둘 모두에 대해, checking_accounts 서브-엔티티는 키 아이콘을 가지는 키 속성 acct_id, 그리고 함께 복합 키를 형성하는 부모 "in" 또는 "out" 최상위 엔티티의 master_account_number 키 속성의 값을 저장하는 또 다른 키 속성을 가진다. 계층 구조 보기 모드에서, 테이블 아이콘들이 그 속성들과 서브-엔티티들(있는 경우)을 보이거나 숨기기 위해 그 엔티티들을 펼치거나 접기 위한 삼각형과 함께 디스플레이된다.

[0048] "엔티티 보기 모드(entity view mode)"에서, 계층 구조의 상이한 레벨들에서 엔티티들에 대한 테이블 아이콘들이 서로 동일한 양으로 들여쓰기 되어 디스플레이되고, 부모 엔티티의 키 속성을 참조하는 키 속성들이 자식 엔티티에서 보여진다. 도 3b는 엔티티 보기 모드에서 입력 섹션(302A)와 출력 섹션(302B) 둘 모두를 나타낸다. 입력 섹션(302A)에 대해, checking_accounts 엔티티와 savings_accounts 엔티티에 대한 테이블 아이콘들은 "in" 엔티티에 대한 아이콘 아래에, 그리고 그것과 수평으로 정렬되어 나타난다. 출력 섹션(302B)에 대해, account_holders 엔티티와 checking_accounts 엔티티와 savings_accounts 엔티티에 대한 테이블 아이콘들이 "out"엔티티에 대한 아이콘 아래에, 그리고 그것과 수평으로 정렬되어 나타난다. 엔티티 보기 모드에서, 부모 엔티티의 키의 (외래 키) 값을 저장하는 키 속성의 존재가 명백하게 보여진다(예를 들어, “in.master_account_number"와 "out.master_account_number" 라고 명명된 필드들). 엔티티 보기 모드에서, 테이블 아이콘들이 그 속성들을 보이거나 숨기기 위해 그 엔티티를 펼치거나 접기 위한 삼각형과 함께 디스플레이되나, 임의의 서브-엔티티들이 독립적으로 펼쳐지거나/접힌다.

[0049] 도 3a와 3b 둘 모두에서 도시된 바와 같이, 소스에서 타깃으로의 매핑 섹션(304)은 소스와 타깃 사이의 매핑들을 정의하기 위한 라인 번호들(308)에 의해 라벨 붙여진 라인들을 포함한다. 상기 매핑들은 임의의 순서로 입력될 수 있고, 사용자는 정의되는 매핑들의 유형들을 기술하는 코멘트들을 제공하기 위해 선택적으로 일부 라인들을 사용할 수 있다. 매핑 정의의 일부로, 사용자는 출력 계층 구조의 엔티티들의 어떤 속성들이 엔티티들의 상이한 인스턴스들을 고유하게 식별하기 위한 키 속성들이 될 것인지를 표시한다. 매핑 모듈(106)은 키 속성들의 이러한 표시에 기반하여, 하기에 더 상세히 설명된 바와 같이 어떤 매핑들이 "매핑들(mappings)"이고 어떤 매핑들이 "집계된 매핑들(aggregated mappings)"인지를 결정한다. 직선 매핑들(straight mappings)에 대해서, 입력 계층 구조의 엔티티의 인스턴스와 출력 계층 구조의 대응하는 엔티티의 인스턴스 사이에 기본 일대일 관계가 있다. 그러나, 하기에 더 상세히 기술된 바와 같이 예를 들어, 입력 엔티티의 일부 인스턴스들이 그것들이 대응하는 출력 엔티티의 인스턴스로서 나타나지 않도록 필터링된다면, 반드시 항상 일대일 관계가 있지는 않고, 대응하는 엔티티들이 반드시 동일한 속성들 또는 서브-엔티티들 모두를 가지진 않는다. 집계된 매핑에 대해, 실행 모듈(112)은 하기에 더 상세히 설명된 바와 같이 입력 엔티티들 및/또는 임시 엔티티들의 측면에서 출력 엔티티의 인스턴스들을 생성하는 프로세스에서 매핑 모듈(106)에 의해 명시된 바와 같이 하나 이상의 집계 연산들을 수행할 것이다. 집계된 매핑들에 대해, 입력 계층 구조의 엔티티의 인스턴스와 출력 계층 구조의 대응하는 엔티티의 인스턴스 사이에 일반적으로 일대일 관계가 없다.

[0050] 소스에서 타깃으로의 매핑 섹션(304)은 사용자가 소스로서 입력 계층 구조로부터의 입력 엔티티 또는 임시 엔티티를 식별하기 위한 소스 열(310), 그리고 사용자가 타깃으로 출력 계층 구조로부터의 출력 엔티티 또는 임시

엔티티를 식별하기 위한 타깃 열(312)을 포함한다. 임시 엔티티는 예를 들어, 타깃으로 정의되었으나, 출력 계층 구조 내에 포함되지 않는 것일 수 있다. 사용자가 필터링되고 매핑된 타깃의 레코드로 전달되지 않는 소스의 특정 레코드들을 식별하는 필터링 함수를 적용하는 선택적 필터를 정의할 수 있도록 하는 필터 열(314)이 있다. 각자 각각 소스와 타깃 엔티티의 인스턴스들에 대응하는 레코드들의 수를 제공하는 레코드 카운트 열들(record count columns)(316A와 316B)이 있다. 사용자가 각각 대응하는 소스 또는 타깃의 인스턴스들(즉, 레코드들)의 보기를 탐색하기 위해 상호작용할 수 있는 아이콘들을 제공하는 보기 열들(318A와 318B)이 있다.

[0051] 도 3c와 3d는 소스에서 타깃으로의 매핑 섹션(304)의 특정 라인에서 식별된 소스와 타깃 사이의 매핑을 정의하기 위한 사용자 인터페이스(320)의 예들의 스크린 샷들을 보여준다. 사용자는 예를 들어, 특정 라인에 대한 매핑 열(319)의 아이콘을 선택하여 이 사용자 인터페이스(320)를 탐색할 수 있다. 도 3c에서, 스크린샷은 “in.checking_accounts"에서 "out.checking_accounts"으로의 매핑(소스에서 타깃으로의 매핑 섹션(304)의 라인 4에 대해)을 보여준다. 점 표기법(dot notation)은 접두부(prefix)로서 엔티티 이름을 가지는 속성 또는 서브-엔티티가 속하는 엔티티를 명백하게 표시하기 위해 특정 상황에서 사용된다. 일부 상황에서, 속성 또는 서브-엔티티가 속하는 엔티티에 관해 모호함이 없다면, 그 속성 또는 서브-엔티티의 이름이 접두부없이 디스플레이될 (또는 입력으로 수신될) 수 있다. 입력 섹션(322)은 표현식/규칙(Expression/Rule) 열(324)에 사용자에 의해 입력된 표현식들에서 사용될 입력들로서 가용한 엔티티들과 그 속성들을 열거한다. 출력/내부 이름(Output/Internal Name) 열(326)은 표현식/규칙 열(324)에서 각각의 표현식에 의해 계산될 출력 엔티티 out.checking_accounts의 각각의 속성을 별도 라인들에서 포함한다. 이 예는 입력 엔티티 in.checking_accounts의 대응하는 인스턴스와 동일한 값을 가지는 것으로 정의될 출력 엔티티 out.checking_accounts의 인스턴스의 5개의 속성들을 포함한다. 특히, 다음 속성들이 정의된다: out.master_account_number(부모 엔티티 "out"의 대응하는 속성의 값을 참조하는 외래 키), out.checking_accounts.acct_id, out.checking_accounts.SSN, out.checking_accounts.balance, 및 out.checking_accounts.interest_rate. in.checking_accounts 엔티티의 대응하는 속성들이 표현식/규칙 열(324)에 홀로 열거된다(앞에 "in" 접두부가 입력 계층 구조로부터 있을 것으로 가정되는 이러한 속성 이름들에 대해 필요로 되지 않는다). 그것은 이 특정 예에서, out.checking_accounts 엔티티의 대응하는 속성들로 정의되지 않은 in.checking_accounts 엔티티의 두 개의 다른 속성들을 남긴다: checking_accounts.first_name과 checking_accounts.last_name. 사용자 인터페이스(320)는 또한 각 라인에서 정의된 대응하는 출력 속성의 값을 보여주는 계산된 값 열(Computed Value column)(328)을 포함한다. 그 출력 속성들이 계산되는 입력 속성들의 값들이 또한 그 속성을 나타내는 필드의 이름 뒤의 괄호 안에, 입력 섹션(322)에서 보여진다. 유형 열(type column)(330)은 그 라인에서 정의된 매핑이 "단순 매핑(simple mapping)"(화살표 아이콘) 또는 "복잡한 매핑(complex icon)"(점선 아이콘)인지를 표시하는 아이콘을 보여준다. 단순 매핑은 동일한 이름의 입력 속성에 출력 속성을 매핑하거나 출력 속성에 상수 값을 할당하는 것이다. 모든 다른 매핑들이 복잡한 매핑들이다. 사용자 인터페이스(300)의 라인에 대해 매핑 열(319)은 그 대응하는 사용자 인터페이스(320)에서 정의된 매핑들 모두가 단순 매핑들이라면 단순 매핑 아이콘을 가지고, 그 대응하는 사용자 인터페이스(320)에서 정의된 매핑들 중 임의의 것이 복잡한 매핑이면 복잡한 매핑 아이콘을 가진다.

[0052] 도 3d에서, 스크린 샷은 “in.checking_accounts"에서 "account_holders"으로의 매핑(소스에서 타깃 매핑 섹션(304)의 라인 2)을 보여준다. 이 매핑에 대한 출력/내부 이름 열(326)은 표현식/규칙 열(324)의 각각의 표현식에 의해 계산될 출력 엔티티 out.account_holders의 각각의 속성을 별도 라인들에서 포함한다. 이 예는 정의될 출력 엔티티 out.account_holders의 5개의 속성들을 포함한다. 다섯 개의 속성들 중 네 개는 입력 엔티티의 인스턴스들의 대응하는 속성들(즉, 동일한 필드 이름)의 측면에서 정의된 출력 엔티티의 인스턴스들의 속성들을 가지는 단순 매핑들이다. 다섯 개의 속성들 중 하나는 잠재적으로 여러 입력 엔티티들의 인스턴스들의 속성들의 측면에서 (out.account_holders 엔티티의 인스턴스들에 대해) 속성 out.account_holders.balance를 정의하는 복잡한 매핑이다. 이 예에서, out.account_holders.balance에 대한 표현식/규칙 열(324)의 표현식은 다음과 같다.

```
sum(in.checking_accounts.balance,in.checking_accounts.SSN) +
sum(in.savings_accounts.balance,in.savings_accounts.SSN==in.checking_accounts.S
SN)
```

[0053]

[0054] 이 표현식은 실행 모듈(112)이 출력 계층 구조의 출력 엔티티들의 인스턴스들을 생성할 때 수행될 집계 연산을 정의한다. 상기 집계 연산은 다음 구문을 가지는 합계 함수(sum function)를 사용하여 정의되는 합이다:

sum(<aggregation_attr>,<match_attr>==<key_attr>). 이 함수의 첫번째 인수 “<aggregation_attr>"는 합에서 피가수(summand)가 될 속성이다. 상기 합산은 인수 엔티티 또는 엔티티들(즉, 그 속성이 인수 <aggregation_attr>로 제공되는 임의의 엔티티)의 여러 인스턴스들에 걸쳐 발생한다. 이 함수의 두번째 인수 “<match_attr>==<key_attr>"는 첫번째 피가수 인수가 합에 포함되기 위해 충족되어야만 하는 조건을 표시하는 표현식 그 자체이다. 키 속성 <key_attr>은 매핑에서 사용되는 입력 엔티티의 키 속성이고, 속성 <match_attr>은 그 키 속성에 매치될 인수 엔티티의 “매치 속성(match attribute)"이다. 이 합계 함수는 속성<match_attr>이 <key_attr>와 동일한 특별 경우에 홀로 열거되는 것을 허용하는 선택적 구문을 가진다. 물론, 사용자는 동일한 효과를 가지는 역 순 “<key_attr >==<match_attr>"으로 표현식을 입력할 수 있다. 그래서, 상기 표현식에 대해, 수행되는 집계는 in.checking_accounts 엔티티 또는 the in.savings_accounts 엔티티 중 하나의 모든 인스턴스들의 “balance" 속성의 값들을 찾아 그 각각의 인스턴스들의 SSN 속성이 동일하다면 그것들을 함께 더한다. 이는 SSN의 값을 그 out.account_holders.SSN 속성으로 가지는 out.account_holders 엔티티의 인스턴스의 out.account_holders.balance 속성에 할당되는 SSN의 각각의 고유 값에 대한 하나의 합산된 총 결과를 산출한다.

[0055] 이 예에서, 출력 계층 구조의 출력 엔티티들의 인스턴스들을 생성하는 상기 실행 모듈(112)의 결과는 집계 연산이 5개의 최상위 "in" 레코드들 중에서 찾아졌던 8개의 in.checking_accounts 레코드들과 4개의 예금 계좌(savings accounts) 레코드들 중에서 SSN 속성의 9개의 고유 값들을 찾은 것을 표시하는 9 개의 out.account_holders 레코드들을 산출한다. 사용자에 의해 정의된 매핑들을 수행한 결과로 생성된 레코드들의 수가 사용자가 생성된 레코드들의 수가 예상된 바와 같았는지를 결정하고, 입력된 표현식들이 정확했는지를 검증하게 하도록 소중한 피드백을 제공하는 출력 섹션(302B)내에서 디스플레이된다. 각각의 엔티티에 대한 레코드들의 총수뿐만 아니라, 다양한 계층 구조 통계(예를 들어, 최소 및 최대 값들)가 계산되고 입력 계층 구조와 출력 계층 구조 둘 모두에 대해 사용자 인터페이스(300)에서 디스플레이될 수 있다. 필터들이 사용된다면, 상기 필터에 의해 거부되고/되거나 허용된 레코드들의 수가 디스플레이될 수 있다.

[0056] 일부 구현에서, 사용자 인터페이스(320)는 필드들과 관련된 이름들(예를 들어, 비즈니스 이름들, 기술 이름들) 사이의 유사성 분석, 및/또는 키 필드들 간의 분석에 기반하여 자동으로 생성되는 입력 엔티티의 필드들과 출력 엔티티의 필드들 사이의 기본 매핑으로 시작할 수 있다. 사용자는 있는 경우 기본 매핑들 중 어떤 것을 받아들일지 결정을 할 수 있거나 자동 매핑 기능(automatic mapping feature)을 해제할 수 있다. 상기 자동 매핑 기능은 사용자가 모든 필드들에 대한 매핑들을 수동적으로 제공해야만 하는 것을 하지 않아도 되게 할 수 있고, 대신 특정 관심 필드들에 대해 매핑들을 제공하는데 초점을 맞춘다.

[0057] 일부 구현에서, 상기 실행 모듈(112)은 출력 레코드들(즉, 출력 계층 구조의 출력 엔티티들의 인스턴스들)을 생성하기 위해 입력 레코드들(즉, 입력 계층 구조의 입력 엔티티들의 인스턴스들)을 처리하기 위해 매핑 모듈(106)에 의해 생성되는 데이터플로 그래프를 실행한다. 도 4는 자동적으로 이러한 데이터플로 그래프들을 생성하는 매핑 모듈(106)에 의해 사용되는 절차(400)의 일 예를 도시한다. 상기 절차(400)는 도 5에서 도시된 예시 데이터플로 그래프(500)의 생성의 설명에서 하기에 더 상세히 설명된 데이터플로 그래프를 구축하는 것과 관련된 상이한 단계들을 포함한다. 상기 절차(400)의 다른 예들이 다른 순서로 동일한 단계들을 수행할 수 있고, 다른 루핑 배열(looping arrangement)을 사용할 수 있거나, 다른 순서로 데이터플로 그래프들(또는 그 등가물)을 구축하는 다른 단계들을 포함할 수 있다.

[0058] 상기 절차(400)는 입력 계층 구조에서 엔티티들의 인스턴스들을 나타내는 레코드들을 저장하는 입력 데이터세트를 나타내는 입력 컴포넌트, 및 출력 계층 구조에서 엔티티들의 인스턴스들을 나타내는 레코드들을 저장하는 출력 데이터세트를 나타내는 출력 컴포넌트를 제공하는 단계(402)를 포함한다. 상기 절차(400)는 또한 입력 컴포넌트에 결합된 분할 컴포넌트(spilt component)와 출력 컴포넌트에 결합된 결합 컴포넌트(combine component)를 제공하는 단계를 포함한다. 분할 컴포넌트는 또 다른 엔티티의 인스턴스의 데이터 구조 내에 임베드된(embedded) 서브-엔티티들의 인스턴스들을 나타내는 임의의 레코드들(또는 다른 벡터 데이터 구조들)을 추출하도록 구성된다. 상기 매핑 모듈(106)은 매핑들의 입력 속성들에 기반하여 분할 컴포넌트를 구성한다. 그래서, 분할 컴포넌트의 출력 포트들 중 적어도 일부는 매핑들 중 하나에서 소스로 사용되는 입력 엔티티의 인스턴스들을 나타내는 레코드들의 흐름을 제공한다. 다른 레코드들 내에 네스트된(nested) 임의의 레코드들이 추출되어서, 더 낮은 레벨 엔티티의 인스턴스를 나타내는 레코드가 그 부모 레코드로부터 제거되고, 더 높은 레벨 엔티티의 인스턴스를 나타내는 레코드가 어떠한 임베드된(embedded) 자식 레코드들도 포함하지 않는다. 결합 컴포넌트는 서브-엔티티의 인스턴스들을 나타내는 임의의 레코드들을 더 높은 레벨 엔티티의 인스턴스의 데이터 구조로 삽입하여 분할 컴포넌트의 역 프로세스를 수행하도록 구성된다. 상기 매핑 모듈(106)은 매핑들의 출력

속성들에 기반하여 결합 컴포넌트를 구성한다.

[0059] 상기 절차(400)는 결합 컴포넌트로의 입력들이 처리되는 외부 루프(406), 그리고 분할 컴포넌트의 출력들이 처리되는 내부 루프(408)를 가진다. 외부 루프(406)에 대한 루프 조건(410)은 처리될 필요가 있는 결합 컴포넌트에 대한 임의의 추가 입력 포트들이 있는지를 결정하고, 입력 포트들의 수는 일반적으로 루트 레벨 바로 아래 출력 계층 구조의 최고 레벨에 대해 생성되는 출력 엔티티들의 수에 기반한다. 외부 루프(406)에서, 매핑 모듈(106)은 각각의 출력 엔티티를 매핑하기 위해 입력들로서 사용될 분할 컴포넌트의 출력들의 수에 관계없이 필요로 되는 데이터플로 그래프의 임의의 컴포넌트들을 생성한다(410). 내부 루프(408)에서, 상기 매핑 모듈(106)은 매핑들에 대한 입력들로 제공되는, 분할 컴포넌트들의 각각의 출력에 대해 다양한 계산들을 수행하기 위해 필요로 되는 데이터플로 그래프의 임의의 컴포넌트들을 생성한다(412). 상기 설명된 바와 같이, 매핑된 입력 속성들(즉, 출력 엔티티의 키 속성들에 매핑되는 것들)이 상기 입력 엔티티의 키 속성들 모두보다 더 적은 것들을 포함하는 각각의 매핑에 대해, 적어도 하나의 컴포넌트가 매핑된 입력 속성들에 대해 동일한 값들을 공유하는 입력 엔티티의 여러 인스턴스들을 집계하기 위해 집계 연산을 수행한다. 다른 컴포넌트들은 또한 분할 컴포넌트에 의해 제공되는 레코드들의 입력 속성들의 특성들에 따라 필요로 될 때 포함될 수 있다.

[0060] 도 5는 사용자에 의해 정의되는 매핑들(116)의 로직을 구현하는 매핑 모듈(106)에 의해 생성되고, 그런 다음 출력 데이터를 생성하는 실행 모듈(112)에 의해 실행되는 데이터플로 그래프(500)의 일 예를 도시한다. 상기 데이터플로 그래프(500)는 InputAccounts.dat이라는 입력 계층 구조의 엔티티들의 인스턴스들을 나타내는 레코드들을 저장하는 입력 데이터세트를 나타내는 입력 컴포넌트(502A), 및 OutputAccounts.dat이라는 출력 계층 구조의 엔티티들의 인스턴스들을 나타내는 레코드들을 저장하는 출력 데이터세트를 나타내는 출력 컴포넌트(502B)를 포함한다.

[0061] 매핑 모듈(106)은 입력 컴포넌트(502A)로부터 입력 레코드들을 검색하는 분할 컴포넌트(504)와 출력 컴포넌트(502B)의 출력 레코드들을 저장하는 결합 컴포넌트(506B)를 이용한다. 이 예에서, 분할 컴포넌트(504)는 상기 보여진 DML 입력 레코드 포맷에 따라 포맷된, 필드 값들의 네스트된 벡터들(nested vectors)로서 임의의 더 낮은 레벨 엔티티들의 임베디드 레코드들(embedded records)을 포함하는 최상위 레코드들의 흐름을 그 입력 포트에서 수신한다. 대안적으로, 컴포넌트들의 다른 유형들은 예를 들어, 엔티티들이 데이터베이스 내 테이블들에 대응하고 그 엔티티들의 인스턴스들이 그 테이블들의 행들에 대응하면 데이터베이스를 판독하거나 기록하는 컴포넌트들같이 입력 레코드들을 수신하고 출력 레코드들을 저장하는 데 사용될 수 있다.

[0062] 분할 컴포넌트(504)의 각각의 출력 포트는 매핑들(114) 중 하나의 소스로서 사용되는 입력 엔티티의 인스턴스들을 나타내는 레코드들의 흐름을 제공한다. 다른 레코드들 내에 네스트된 임의의 레코드들이 추출되어, 더 낮은 레벨 엔티티의 인스턴스를 나타내는 레코드가 그 부모 레코드로부터 제거되고, 더 높은 레벨 엔티티의 인스턴스를 나타내는 레코드가 임의의 자식 레코드들을 포함하지 않는다. 매핑 모듈(106)은 매핑들이 직선 매핑들 또는 집계된 매핑들인지를 포함하여, 정의된 특정 매핑들(114)의 구조에 기반하여 분할 컴포넌트(504)에 대해 필요로 되는 출력 포트들의 수를 결정한다. 매핑 모듈(106)은 결합 컴포넌트(506)에 대해 필요로 되는 입력 포트들의 수를 결정한다(이 예에서는 네 개).

[0063] 매핑 모듈(106)은 (출력 계층 구조의 엔티티들 또는 임의의 임시 엔티티들을 포함하여) 적어도 하나의 매핑의 타깃들인 엔티티들에 대해 사용자가 정의한 키 속성들에 기반하여 매핑이 직선 매핑인지 집계 매핑인지를 결정한다. 타깃 엔티티의 각각의 키 속성(함께 기본 키를 구성함)에 대해, 매핑 모듈(106)은 그 매핑의 소스인 엔티티(입력 계층 구조의 엔티티 또는 임시 엔티티)의 대응하는 입력 속성들을 결정한다. 이러한 "매핑된 입력 속성들"은 타깃 엔티티의 키 속성에 직접 매핑될 수 있거나(예를 들어, 단순 매핑), 타깃 엔티티의 키 속성을 결정하기 위한 표현식에서 사용될 수 있다(예를 들어, 복잡한 매핑에서).

[0064] 이러한 매핑된 입력 속성들의 특성들에 따라, 매핑 모듈(106)은 "직선 매핑(straight mapping)" 또는 "집계 매핑(aggregated mapping)"으로 매핑을 분류한다. 매핑 모듈(106)은 매핑된 입력 속성들이 소스 엔티티의 기본 키를 커버(cover)하는지를 결정하기 위해 매핑된 입력 속성들을 소스 엔티티의 하나 이상의 키 속성들(함께 기본 키를 구성함)과 비교한다. 매핑된 입력 속성들이 소스 엔티티의 키 속성들 모두를 포함한다면, 매핑된 입력 속성들은 기본 키를 커버한다. 매핑된 입력 속성들이 소스 엔티티의 키 속성들 모두보다 더 적게 포함한다면, 매핑된 입력 속성들은 기본 키를 커버하지 못한다. 매핑된 입력 속성들이 기본 키를 커버한다면, 상기 매핑이 (특정 타깃 기본 키로) 타깃 엔티티의 각각의 인스턴스에 대해 (특정 소스 기본 키로) 소스 엔티티의 고유 인스턴스를 찾는 것이 보장되고, 상기 매핑은 "직선 매핑"으로 분류된다. 매핑된 입력 속성들이 기본 키를 커버하지 않는다면, 상기 매핑이 타깃 엔티티의 각각의 인스턴스에 대해 소스 엔티티의 고유 인스턴스를 찾는 것이 보장

되고, 상기 매핑은 "집계 매핑"으로 분류된다.

[0065] 매핑된 입력 속성들이 기본 키를 커버하는지 여부를 결정할 때, 어떤 종류의 매핑이 타깃 엔티티의 키 속성과 소스 엔티티의 키 속성 사이에 존재하는가를 결정하는 것이 또한 필수일 수 있다. 매핑이 일대일 매핑이 아니라면(예를 들어, 대신에 다대일 매핑이라면), 하나의 기본 키 값이 또 다른 기본 키 값과 동일한 값으로 매핑할 수 있고, 따라서 타깃 엔티티의 각각의 인스턴스에 대한 소스 엔티티의 고유 인스턴스가 보장되지 않는다. 상기 매핑은 사용자에 의해 제공된 표현식에 의해 정의된 함수 f(x)가 수학적 의미로 일대일이라면 일대일 매핑이다(즉, x != y 는 f(x) != f(y)를 의미하고, 여기서 “!=" 는 같지 않음을 의미함). 상기 매핑이 일대일 매핑이라면, 하나 이상의 매핑된 입력 속성들은 출력 엔티티의 각각의 키 속성들과 일대일 대응을 가진다.

[0066] 집계 매핑에 대해, 집계 연산은 잠재적으로 소스 엔티티의 여러 인스턴스들이 정보(예를 들어, 속성 값들)를 타깃 엔티티의 특정 인스턴스의 계산에 제공하는 것을 허용하도록 수행된다. 타깃 엔티티의 기본 키에 부합하는 소스 엔티티의 단일 인스턴스만 있는 것으로 판명되면, 상기 집계 연산은 단순하게 상기 매핑에서 사용하는 그 하나의 인스턴스로부터 정보를 획득한다. 일부 경우에, 타깃 엔티티의 기본 키에 부합하는 소스 엔티티의 여러 인스턴스들이 있을지라도, 상기 집계 연산은 단순하게 상기 매핑에서 사용되는 인스턴스들 중 단 하나를 선택할 수 있다.

[0067] 이 예에서, 상기 매핑 모듈(106)은 세 개의 직선 매핑들과 두 개의 집계 매핑들을 결정하고, 그 매핑들을 수행하기 위해 필요로 되는 데이터플로 그래프(500)의 컴포넌트들을 생성한다. 하나의 출력 포트는 소스에서 타깃으로의 매핑 섹션(304)의 1번 라인의 직선 매핑을 위해 최상위 "in" 엔티티의 인스턴스들을 나타내는 레코드들을 Map 컴포넌트(512A)에 제공한다. 다른 출력 포트들은 소스에서 타깃으로의 매핑 섹션(304)의 4번과 5번 라인들의 직선 매핑들을 위해 in.checking_accounts와 in.savings_accounts 엔티티들의 인스턴스들을 나타내는 레코드들을 Map-3 컴포넌트(512B)와 Map-4 컴포넌트(512C)로 각각 제공한다. 이러한 직선 매핑들에 대한 컴포넌트들(Map 컴포넌트(512A), Map-3 컴포넌트(512B), 및 Map-4 컴포넌트(512C))은 소스 엔티티의 인스턴스로부터 매핑된 속성 값들을 판독하고 결합 컴포넌트(506)의 포트에서 수신되는 타깃 엔티티의 대응하는 인스턴스로 그 매핑된 속성 값들을 기록하는 동작을 수행한다. 이러한 컴포넌트들은 대응하는 매핑에 대해 정의된 임의의 필터를 선택적으로 적용하도록 구성될 수 있거나, 별도 컴포넌트들이 이러한 필터링을 적용하도록 데이터플로 그래프(500)에 추가될 수 있다. 이러한 세 개의 매핑들이 직선 매핑들인 이유는 출력 엔티티의 기본 키를 형성하는 키 속성들이 입력 엔티티의 전체 기본 키를 함께 형성하는 각각의 키 속성들에 매핑되기 때문이다. 예를 들어, 4번 라인의 매핑에 대해, out.checking_accounts 엔티티의 기본 키는 out.checking_accounts.acct_id와 out.master_account_number 키 속성들로 구성되고, 그것은 in.checking_accounts.acct_id 와 in.master_account_number 키 속성들로 구성된 in.checking_accounts 엔티티의 전체 기본 키로 매핑한다.

[0068] 분할 컴포넌트(504)의 다른 출력 포트들은 소스에서 타깃으로의 매핑 섹션(304)의 2번과 3번 라인의 두 개의 집계된 매핑들에 대해 표현식들에서 참조되어 사용되는 엔티티들의 인스턴스들을 나타내는 레코드들을 제공한다. 이러한 두 개의 매핑들이 집계된 매핑들인 이유는 출력 엔티티의 기본 키를 형성하는 키 속성들이 입력 엔티티의 키 속성들 모두를 포함하지 않는 각각의 속성들에 매핑되기 때문이다. 예를 들어, 2번 라인의 매핑에 대해, out.account_holders 엔티티의 기본 키는 out.account_holders.SSN과 out.master_account_number 키 속성들로 구성되고, in.checking_accounts 엔티티의 기본 키의 키 속성들 중 하나(즉, in.checking_accounts.acct_id 속성)를 포함하지 않는다. 데이터플로 그래프(500)가 특정 집계된 매핑에 대해 집계 연산을 어떻게 수행하는지를 결정하기 위해, 매핑 모듈(106)은 먼저 사용자 인터페이스(320)에서 사용자에 의해 제공되는 표현식들이 집계된 매핑에서 사용되는 소스와 타깃 엔티티들의 속성들에 대해 이러한 집계 연산을 정의하는지를 결정한다. 그렇다면, 상기 매핑 모듈(106)은 데이터플로 그래프(500)에 매핑된 입력 속성들에 대해 동일한 값들을 공유하는 입력 엔티티의 여러 인스턴스들을 집계하기 위해 집계 연산("롤업(rollup)"연산이라고도 함)을 수행하는 롤업 컴포넌트(rollup component)를 추가할 것이다. 사용자에 의해 제공되는 표현식들이 이러한 집계 연산을 정의하는 집계된 매핑에서 사용되는 속성들에 대한 표현식들을 제공하지 않는다면, 상기 매핑 모듈은 데이터플로 그래프(500)에 의해 수행되는 기본 집계 연산을 적용한다. 예를 들어, 여러 인스턴스들 중 마지막 것으로부터의 속성 값들이 사용되는,"중복제거(de-duplication)" 연산이 롤업 컴포넌트에 의해 구현되는 임의의 집계 연산의 일부로서 포함될 수 있다. 각각의 집계 매핑에 대해 이러한 롤업 컴포넌트의 이러한 삽입은 사용자가 소스와 타깃 엔티티의 속성들을 매핑하기 위해 명시적인 집계 연산을 제공하든 안하든, 특정 기본 키를 가지는 타깃 엔티티의 단일 고유 인스턴스가 있을 것을 보장한다.

[0069] 분할 컴포넌트(504)의 출력 포트들은 소스에서 타깃으로의 매핑 섹션(304)의 2번 라인의 집계된 매핑을 위해 in.checking_accounts와 in.savings_accounts 엔티티들의 인스턴스들을 나타내는 레코드들을 Rollup 컴포넌트

(514A)와 Rollup-1 컴포넌트(514B)에 각각 제공한다. 이러한 매핑의 속성들에 대한 표현식들이 두 개의 합산(summations)의 형태로 집계 연산을 포함하는 하나의 표현식(즉, 표현식/규칙 열(324)의 4번 라인)을 포함하기 때문에, 매핑 모듈(106)은 타깃 엔티티의 기본 키를 형성하는 키 속성들에 롤업을 수행하는 합산들 각각에 대해 롤업 컴포넌트를 추가한다. 이 예에서, 타깃 엔티티의 기본 키는 out.account_holders.SSN와 out.master_account_number 속성들로 구성된다. Rollup 컴포넌트(514A)는 이러한 키 속성들에 기반하여 피가수 조건을 만족하는 모든 인스턴스들에 대해 피가수 인수 in.checking_accounts.balance를 추가하여 첫번째 합산을 수행한다. 이 예에서, 출력 엔티티 out.account_holders는 기본 키로 SSN을 포함하나, SSN은 매치 속성(match attribute)이 동일한 SSN 값을 가지는 여러 입력 엔티티 인스턴스들을 찾을 수 있을 때 SSN을 이용하여 정의된 합산을 의미하는 입력 엔티티 in.checking_accounts의 기본 키 속성의 일부가 아니다. Rollup-1 컴포넌트(514B)는 이러한 키 속성들에 기반하여 피가수 조건을 만족하는 모든 인스턴스들에 대해 피가수 인수 in.savings_accounts.balance를 추가하여 두 번째 합산을 수행한다.

[0070] 매핑 모듈(106)은 집계 연산을 완성하기 위해 다른 컴포넌트들을 추가한다. Join 컴포넌트(516A)는 키 속성 값들이 동일한 롤업 컴포넌트들에 의해 수행되는 두 개의 합산들의 결과들을 찾아 더하고, 그 출력 포트의 조인된(joined) 출력 레코드를 Map-1 컴포넌트(512D)로 제공한다. Map-1 컴포넌트(512D)는 조인된 레코드의 두 개의 값들의 합을 수행하고, 그 최종 값과 관련된 키 속성들의 특정 값들과 함께, out.account_holders.balance 속성의 값들로 그 최종 값을 그 출력 포트의 레코드에 제공한다.

[0071] 유사하게, 다른 출력 포트들은 소스에서 타깃으로의 매핑 섹션(304)의 3번 라인의 집계 매핑을 위해 Rollup-3 컴포넌트(514C)와 Rollup-4 컴포넌트(514D) 각각에 in.savings_accounts와 in.checking_accounts 엔티티들의 인스턴스들을 나타내는 레코드들을 제공한다. 이 매핑의 속성들에 대한 표현식들은 또한 두 개의 합산들의 형태로 집계 연산을 포함하는 하나의 표현식을 포함한다. 따라서, 상기 설명된 바와 유사한 동작들을 수행하는 대응하는 롤업 컴포넌트들(Rollup-3 컴포넌트(514C) 및 Rollup-4 컴포넌트(514D)), 및 조인과 맵 컴포넌트들(Join-2 컴포넌트(516B) 및 Map-2 컴포넌트(512E))이 있다.

[0072] 매핑 모듈(106)은 동일한 타깃 엔티티(out.account_holders)에 대한 두 개의 연속적인 매핑들의 결과를 수집하기 위해 (예를 들어, 다른 하나의 흐름로부터의 모든 레코드들 뒤에 하나의 흐름으로부터의 레코드들을 덧붙여, 또는 흐름들 사이에 번갈아 레코드들을 병합하여) 수신된 레코드들의 두 개의 흐름들로부터 레코드들의 단일 흐름을 형성하는, 수집 컴포넌트(gather component)(518)를 데이터플로 그래프(500)에 삽입한다. 매핑 모듈(106)은 또한 두 개의 매핑들에 의해 생성되는 임의의 중복 레코드들을 제거하기 위해 중복제거(deduplicate) 컴포넌트(420)를 삽입한다. 예를 들어, 2번 라인으로부터의 매핑은 동일한 SSN을 가지는 대응하는 예금 계좌들 없이 당좌 예금 계좌들을 찾았을 수 있고, 3번 라인으로부터 매핑은 동일한 SSN을 가지는 대응하는 당좌 예금 계좌들 없이 예금 계좌들을 찾았을 수 있으나, 두 매핑들 모두 동일한 SSN을 가지는 당좌 예금 계들과 예금 계좌들의 쌍을 발견했을 수 있다.

[0073] 일부 매핑들에 대해, 매핑 모듈(106)은 생성된 데이터플로 그래프에 추가 컴포넌트들을 추가할 필요가 있을 수 있다. 예를 들어, 입력 계층 구조의 입력 레벨과 출력 계층 구조의 출력 레벨에 기반하여, 상기 그래프는 명시된 매핑 규칙들로 입력 레코드들의 흐름으로부터 특정 정보를 출력 레코드들의 올바른 필드들로 넣기 위해 다양한 연산들을 미리 형성할 필요가 있을 수 있다. 집계된 매핑에 대해, 롤업 컴포넌트는 관련된 집계 연산을 수행하는 데 필요로 될 수 있으나, 또한 추가 집계 연산들을 수행하는 데 필요로 될 수 있는 다른 롤업 컴포넌트들이 있을 수 있다. 출력 필드의 정보가 두 개의 상이한 입력 필드들로부터의 정보로부터 도출된다면 조인 컴포넌트가 필요로 될 수 있다. 정렬 컴포넌트들(sort components)을 포함하는지를 결정하기 위해 예를 들어, 매핑 모듈(106)은 어떻게 정렬 키들이 정렬 연산(정렬 컴포넌트에 의해 수행됨)이 필요로 되는지 여부와 필요로 되는 곳을 결정하기 위해 매핑되는지를 비교한다. 일부 구현에서, 매핑 모듈(106)은 생성된 데이터플로 그래프를 중복을 줄이기 위해 제거하는 부분들, 또는 덜 효율적이거나 더 효율적인 컴포넌트들로 대체되는 부분들 같은 계산의 특정 부분들을 최적화하기 위해 변경한다. 데이터플로 그래프(500)의 컴포넌트들을 생성하고 그 포트들을 적합하게 연결하는 것뿐만 아니라, 매핑 모듈(106)은 매핑된 출력 데이터를 생성하기 위해 또는 추적 정보를 사용자에게 제공하기 위해 필요로 될 수 있는 다른 데이터 구조들을 생성할 수 있다. 예를 들어 매핑 모듈은 인스턴스들이 생성되었던 입력 엔티티들(즉, 입력 레코드들)의 대응하는 인스턴스들과 그 레코드들과 임의의 중간 레코드들에 수행되는 연산들을 보여주는 출력 엔티티들(즉, 출력 레코드들)의 특정 인스턴스들의 계통 표현들을 생성하는데 사용되는 계통 정보(lineage information)를 저장하도록 구성될 수 있다.

[0074] 이러한 매핑 기술들은 데이터플로 그래프의 일부가 메타프로그램되는(metaprogrammed) (즉, 일부 사용자 정의 제약조건들에 기반하여 자동적으로 생성되는) 상황에서 사용될 수 있다. 하나의 이러한 예에서, 데이터플로 그

래프는 사용자 정의 변환에 따라 입력 데이터를 사용자 정의 입력 포맷으로부터 사용자 정의 출력 포맷으로 변환하기 위해 구성될 것이다. 예를 들어, 본원에서 참조로 원용된"서브 그래프에 대한 인터페이스 관리(MANAGING INTERFACES FOR SUB-GRAPHS)'라는 타이틀의 2014년 12월 5일에 출원된 미국 출원 번호 14/561,435에 기술된 바와 같이 데이터플로 그래프는 서브-그래프 인터페이스를 포함하는 포괄 컨테이너 그래프(generic container graph)를 포함할 수 있다. 상기 서브 그래프 인터페이스는 실행시간 전에 컨테이너 그래프로 삽입되게, 서브 그래프의 특정 구현이 사용자 입력으로부터 적어도 부분적으로 도출되게 하는 것을 가능하게 한다. 실행 시간 직전에, 사용자는 입력 포맷, 출력 포맷, 및/또는 입력 포맷의 필드들과 출력 포맷의 필드들 사이의 매핑들에 관련된 많은 질문들을 받을 수 있다. 상기 질문들에 대한 사용자 응답들에 기반하여, 서브 그래프의 구현은 상기 매핑 기술들을 이용하여 자동적으로 생성된다(즉, 메타프로그램된다).

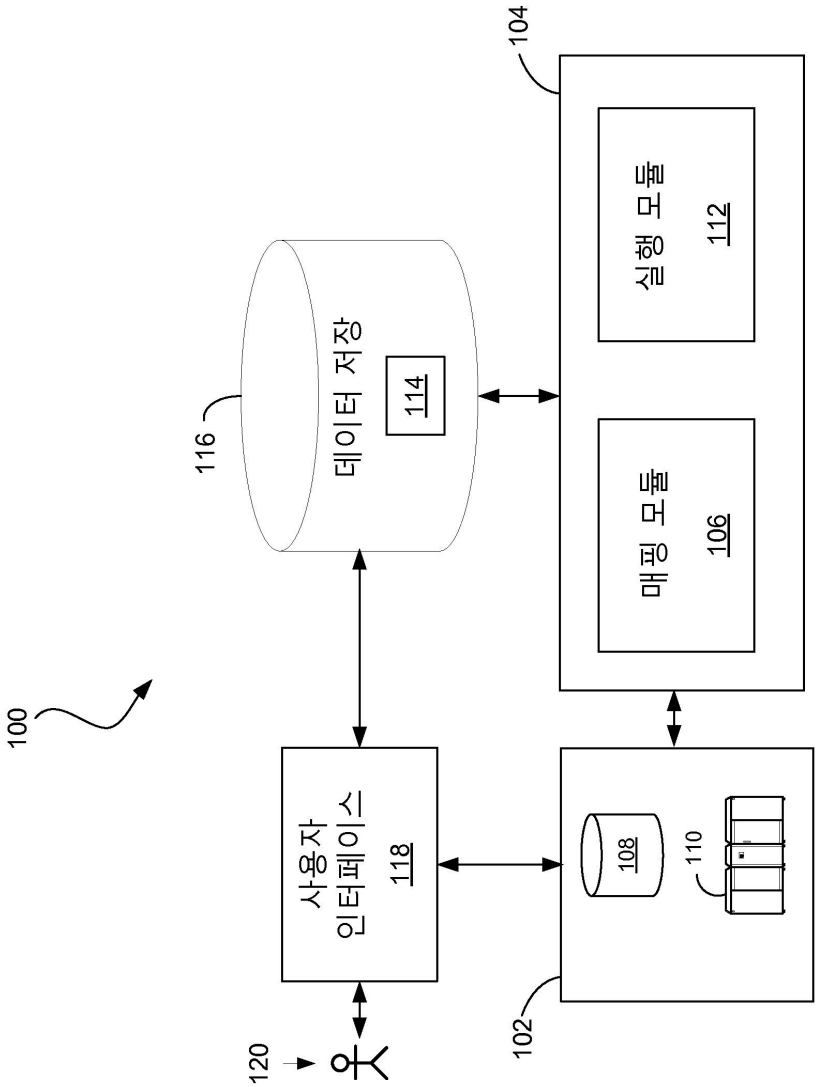
[0075] 상기 설명된 매핑 방법은 예를 들어, 적절한 소프트웨어 명령들을 실행하는 프로그래머블 컴퓨팅 시스템을 이용하여 구현될 수 있거나 필드 프로그래머블 게이트 어레이(FPGA) 같은 적절한 하드웨어, 또는 어떤 하이브리드 형태로 구현될 수 있다. 예를 들어, 프로그램된 방법에서 상기 소프트웨어는 하나 이상의 프로그램되거나 프로그램가능한 컴퓨팅 시스템(분산, 클라이언트/서버, 또는 그리드같은 다양한 아키텍처를 가질 수 있음) - 각각은 적어도 하나의 프로세서, 적어도 하나의 저장 시스템(휘발성 및/또는 비휘발성 메모리 및/또는 저장 요소들을 포함함), 적어도 하나의 사용자 인터페이스(적어도 하나의 입력 디바이스 또는 포트를 이용하여 입력을 수신하고, 적어도 하나의 출력 디바이스 또는 포트를 이용하여 출력을 제공하기 위함)를 포함함 - 상에서 실행하는 하나 이상의 컴퓨터 프로그램들에서의 절차들을 포함할 수 있다. 상기 소프트웨어는 예를 들어, 데이터플로(dataflow) 그래프들의 디자인, 구성, 및 실행에 관련된 서비스들을 제공하는 더 큰 프로그램의 하나 이상의 모듈들을 포함할 수 있다. 상기 프로그램의 모듈들(예를 들어, 데이터플로 그래프의 요소들)은 데이터 저장소에 저장된 데이터 모델에 부합하는 데이터 구조들 또는 다른 구조화된 데이터로 구현될 수 있다.

[0076] 상기 소프트웨어는 (예를 들어 범용 또는 특별 목적 컴퓨팅 시스템 또는 디바이스에 의해 판독 가능한) CD-ROM 또는 다른 컴퓨터 판독 가능 매체 같은 유형의, 비 일시적인 매체 상에서 제공될 수 있거나 또는 그것이 실행되는 컴퓨팅 시스템의 유형의, 비 일시적 매체에 네트워크 통신 매체를 통해 (예를 들어, 전파 신호에 인코딩되어) 전달될 수 있다. 상기 처리의 일부 또는 전부는 특별 목적 컴퓨터 상에서 또는 코프로세서(coprocessors) 또는 필드 프로그래머블 게이트 어레이(field-programmable gate array, FPGA), 또는 전용, 주문형 반도체(dedicated, application-specific integrated circuit, ASIC) 같은 특별 목적 하드웨어를 사용하여 수행될 수 있다. 상기 처리는 소프트웨어에 의해 명시된 계산의 상이한 부분들이 상이한 컴퓨팅 요소들에 의해 수행되는 분산 방식으로 실행될 수 있다. 각각의 이러한 컴퓨터 프로그램은 저장 장치 매체가 본 출원에서 기술된 처리를 수행하도록 컴퓨터에 의해 판독될 때 컴퓨터를 구성하고 작동하기 위해, 바람직하게는 범용 또는 특별 목적 프로그래머블 컴퓨터에 의해 액세스 가능한 저장 장치의 컴퓨터 판독 가능 저장 매체(예를 들어, 솔리드 스테이트 메모리(solid state memory) 또는 매체, 또는 자기 또는 광 매체)에 저장되거나 다운로드된다. 본 발명 시스템은 또한 컴퓨터 프로그램으로 구성되는, 유형의, 비 일시적 매체로서 구현되는 것으로 간주될 수 있고, 그렇게 구성된 매체는 컴퓨터가 본 출원에서 기술된 처리 단계들 중 하나 이상을 수행하도록 구체적이고 미리 정의된 방식으로 동작하게 한다.

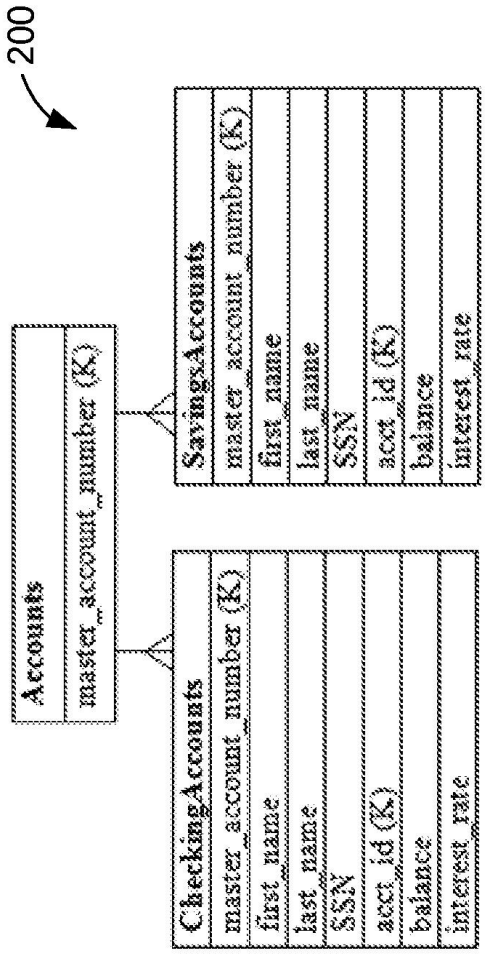
[0077] 본 발명의 다수의 실시 예가 기술되었다. 그럼에도 불구하고, 전술한 설명은 예시를 위한 것이며 다음의 청구항들의 범위에 의해 정의되는 본 발명의 범위를 한정하는 것이 아닌 것으로 이해되어야 한다. 따라서 다른 실시예들이 또한 다음 청구항들의 범위 내에 있다. 예를 들어, 다양한 변형이 본 발명의 범위를 벗어남 없이 만들어질 수 있다. 부가적으로, 전술된 단계들의 일부는 순서 독립적이므로 기술된 것과 다른 순서로 수행될 수 있다.

도면

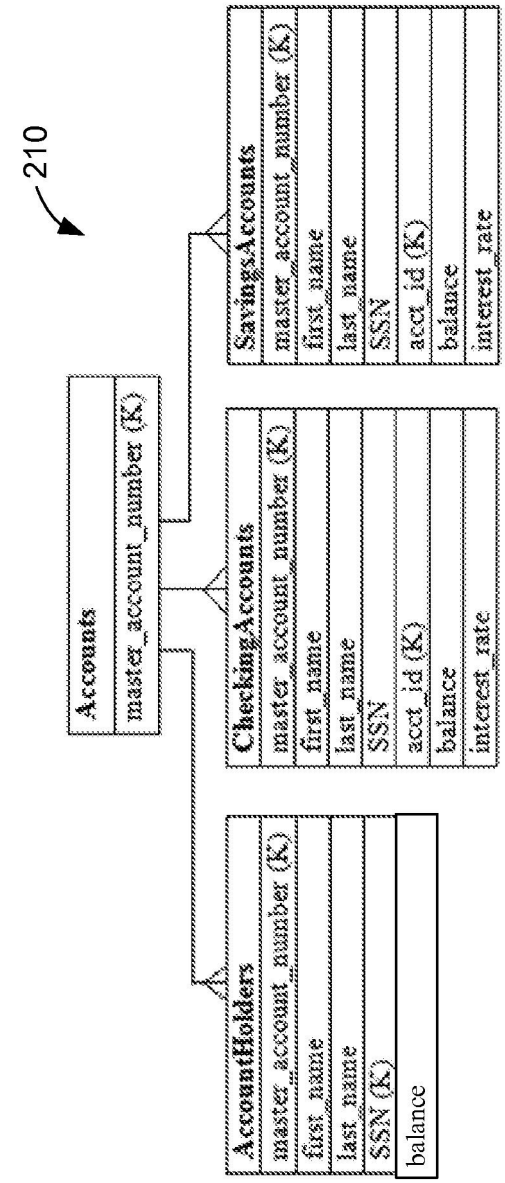
도면1


100
104
실행 모듈
112
매핑 모듈
106
116
데이터 저장
114
사용자
인터페이스
118
108
110
102
120

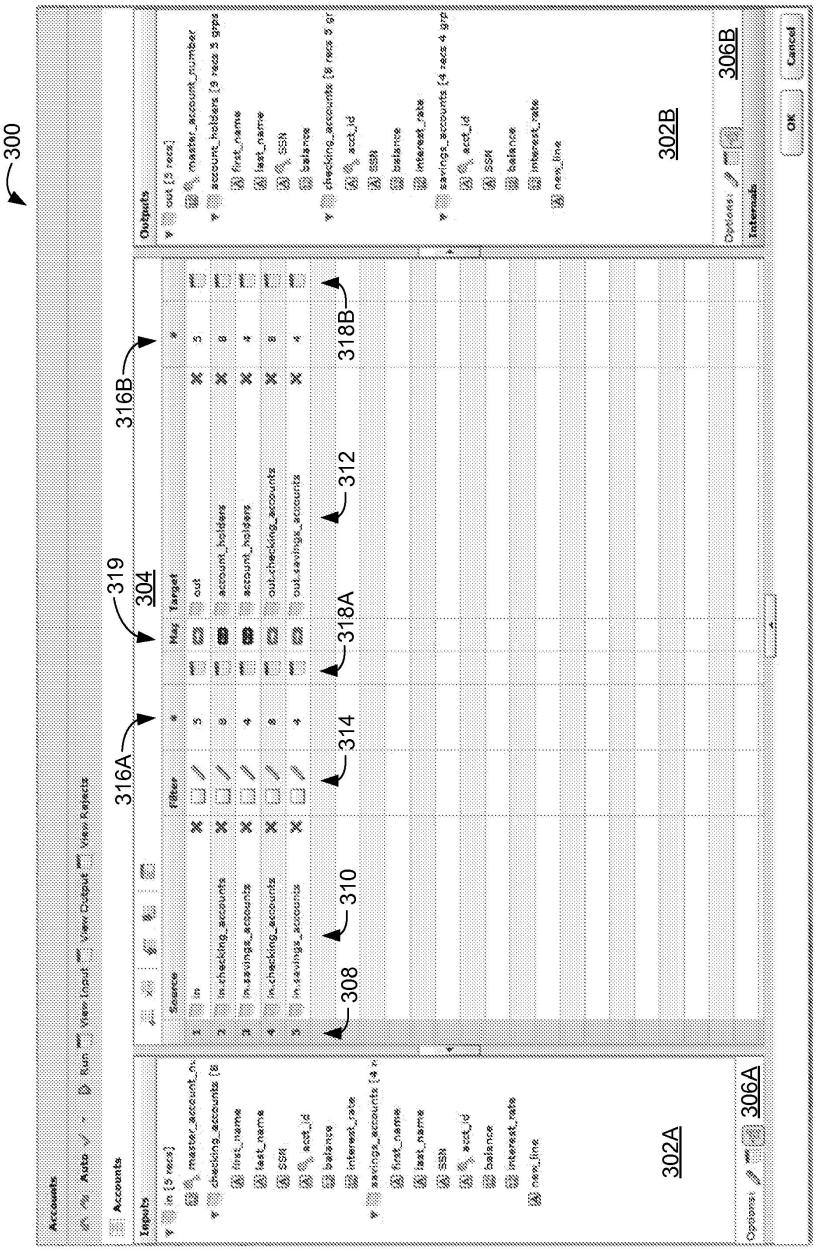
도면2a


200
Accounts
master_account_number (K)
SavingsAccounts
master_account_number (K)
first_name
last_name
SSN
acct_id (K)
balance
interest_rate
CheckingAccounts
master_account_number (K)
first_name
last_name
SSN
acct_id (K)
balance
interest_rate

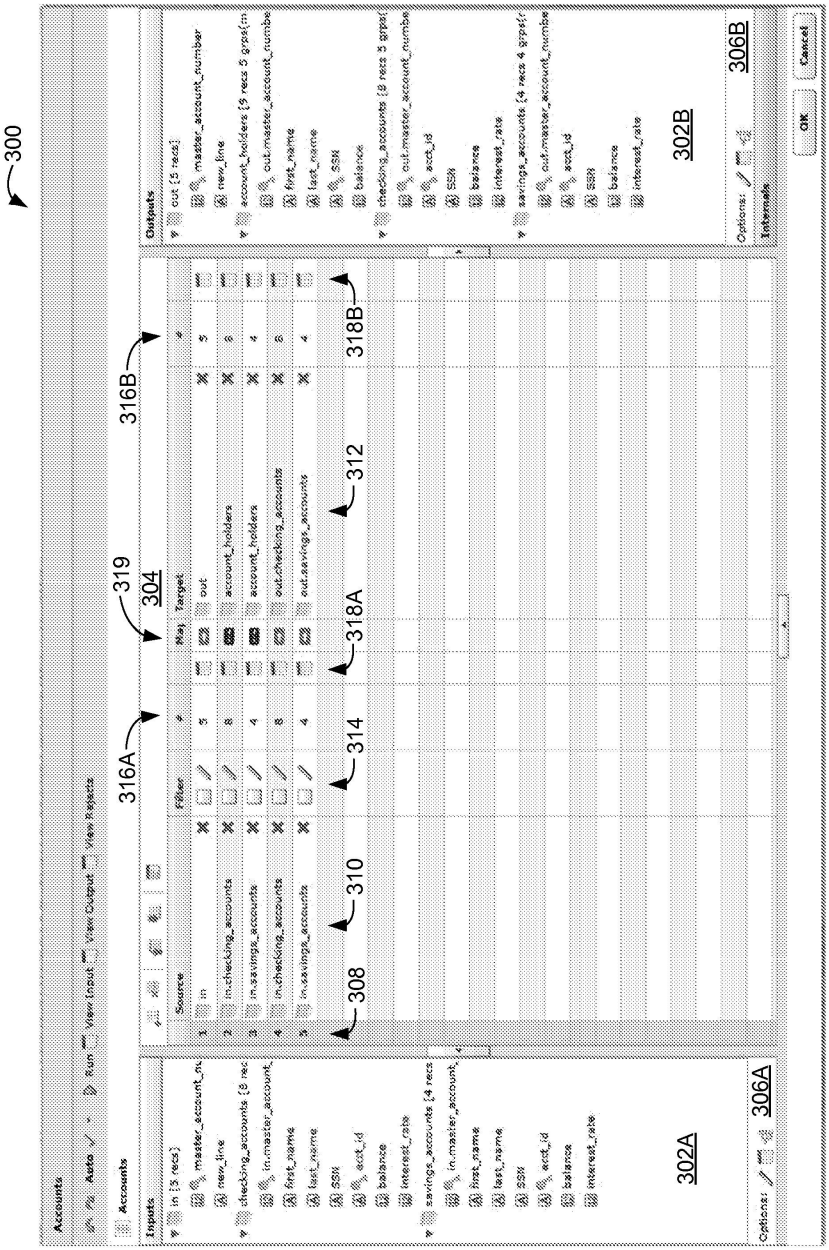
도면2b


210
Accounts
master_account_number (K)
SavingsAccounts
master_account_number (K)
first_name
last_name
SSN
acct_id (K)
balance
interest_rate
CheckingAccounts
master_account_number (K)
first_name
last_name
SSN
acct_id (K)
balance
interest_rate
AccountHolders
master_account_number (K)
first_name
last_name
SSN (K)
balance

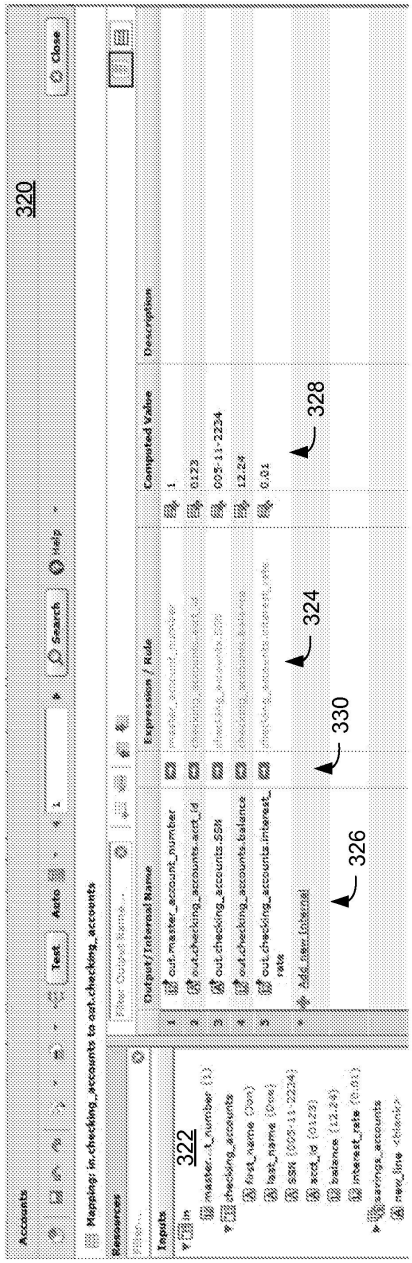
도면3a


300
Accounts
Auto
Run
View Input
View Output
View Rejects
Accounts
Inputs
Source
Filter
Map
Target
Outputs
in
out
master_account_number
checking_accounts
account_holders
first_name
last_name
SSN
balance
savings_accounts
acct_id
interest_rate
new_line
checking_accounts
savings_accounts
Options:
Intervals
OK
Cancel
302A
302B
304
306A
306B
308
310
312
314
316A
316B
318A
318B
319

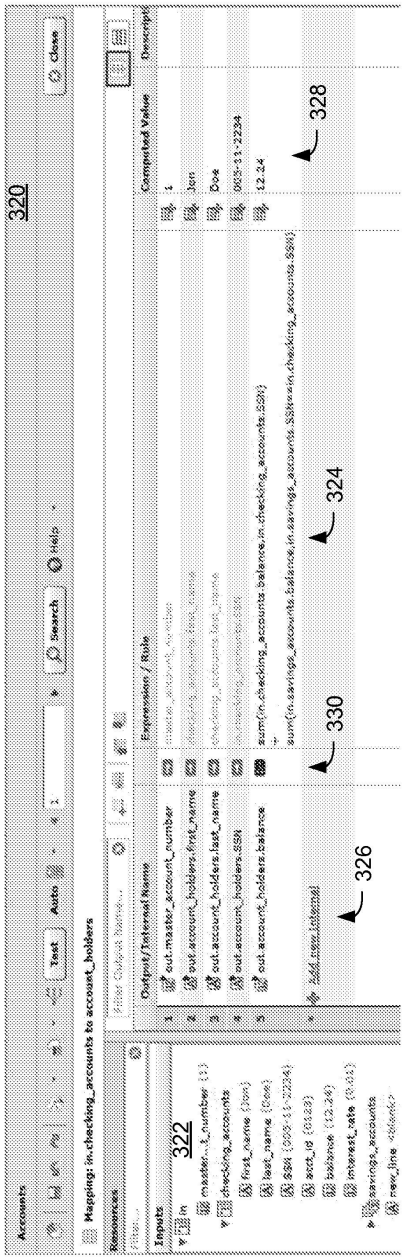
도면3b


300
316A
319
304
316B
308
310
314
318A
312
318B
302A
302B
306A
306B

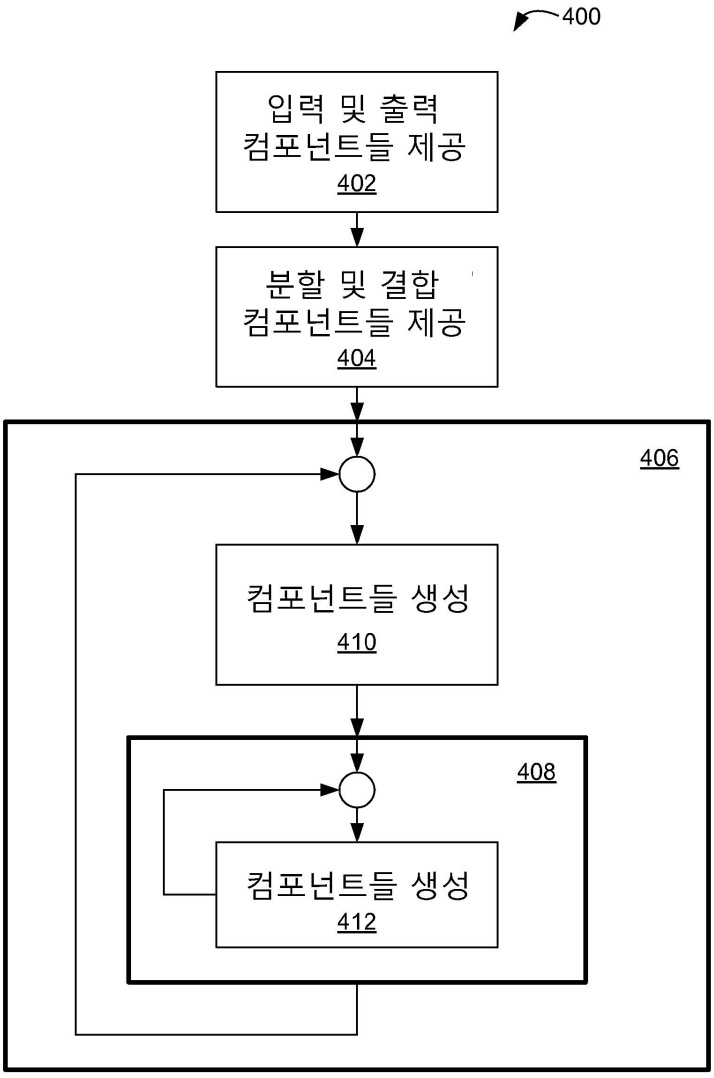
도면3c



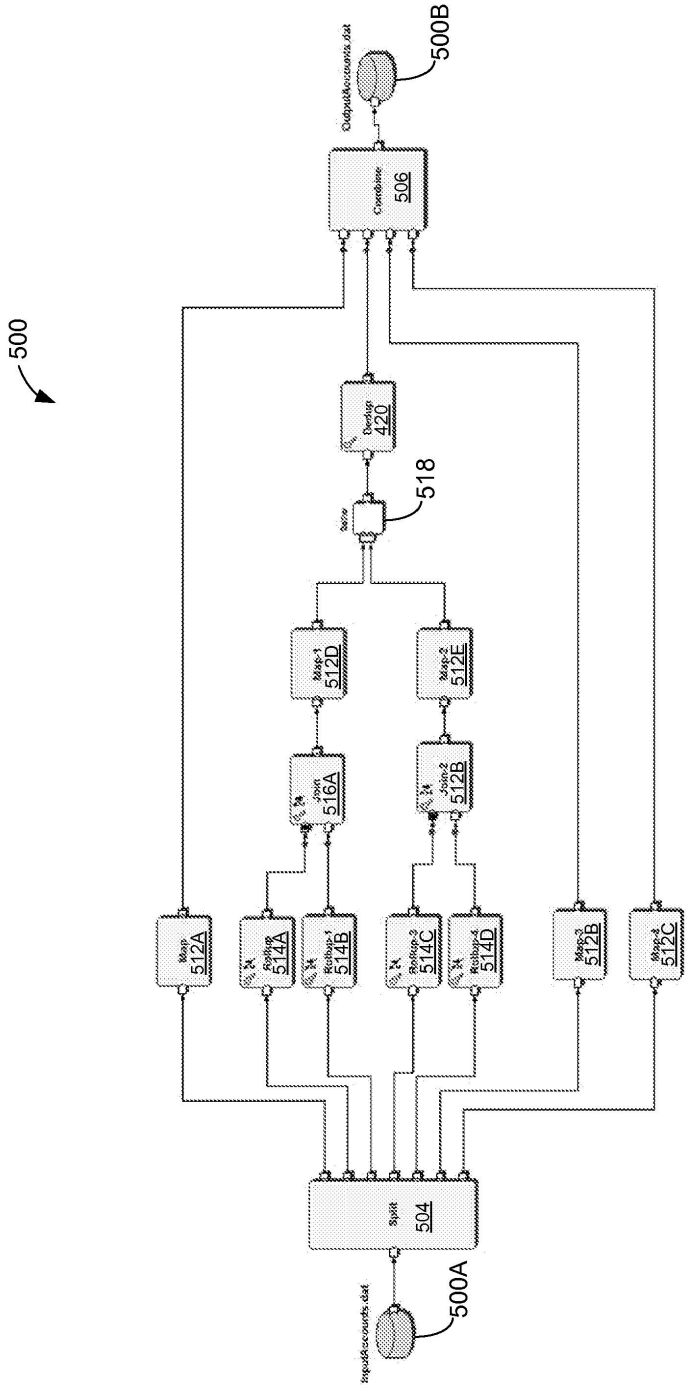
도면3d



도면4


400
입력 및 출력 컴포넌트들 제공
402
분할 및 결합 컴포넌트들 제공
404
406
컴포넌트들 생성
410
408
컴포넌트들 생성
412

도면5


500
500B
506
420
518
512D
512E
516A
512B
512A
514A
514B
514C
514D
512B
512C
504
500A