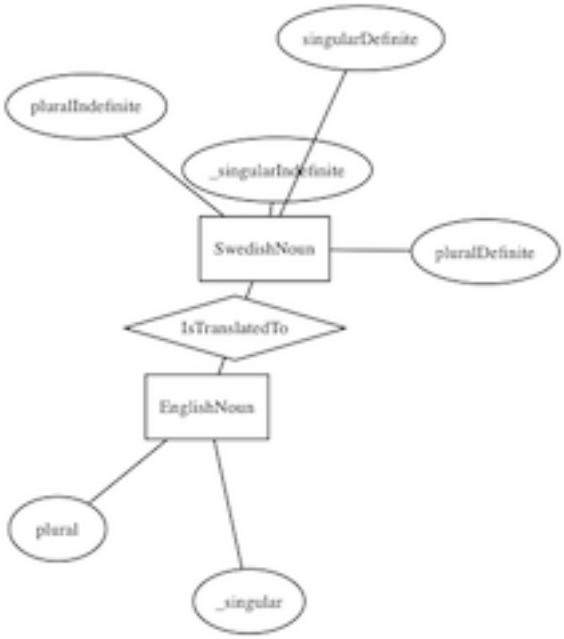
Databases Re-exam August 2015

Solutions

Aarne Ranta

1 ER diagrams and schemas




```
EnglishNoun(_singular,plural)

SwedishNoun(_singularIndefinite,singularDefinite,pluralIndefinite,pluralDefinite)

IsTranslatedTo(_englishNounSingular,_swedishNounSingularIndefinite)
  englishNounSingular -> EnglishNoun.singular
  swedishNounSingularIndefinite -> SwedishNoun.singularIndefinite
```

2 Functional dependencies and normal forms

From lecture notes, section 4.2.4:

country	product	exportTo
Sweden	cars	Norway
Sweden	paper	Denmark
Sweden	cars	Denmark
Sweden	paper	Norway

```
key (country,product,exportTo)
MVD country ->> product
```

Not in 4NF. Decomposition:

country	product
Sweden	cars
Sweden	paper

```
key (country,product)
```

country	exportTo
Sweden	Norway
Sweden	Denmark

```
key (country,exportTo)
```

3 SQL DDL and Queries

```
-- 3a
CREATE TABLE Words (
  string text,
  lemma text,
  class text,
  description text,
  constraint class_names check (class in ('noun','verb','adjective')),
  constraint words_prim_key primary key (lemma,class,description)
  ) ;

-- 3b

SELECT lemma, string
FROM Words
WHERE description LIKE 'plural indefinite %' ;

-- 3c

SELECT class, description, COUNT(string) AS occurrences
FROM Words
GROUP BY class, description
ORDER BY COUNT(string) DESC ;
```

4 Relational algebra

$$\pi_{A.string}\sigma_{A.string=B.stringANDA.class<B.class}(\rho_A words \times \rho_B words)$$

5 Triggers

```
-- 5a

CREATE VIEW Verbs AS (
  SELECT I.string AS infinitive, Pr.string AS present,
           P.string AS past, S.string AS supine
  FROM Words I, Words Pr, Words P, Words S
  WHERE
    I.lemma = Pr.lemma AND I.lemma = P.lemma AND I.lemma = S.lemma AND
    I.description = 'infinitive' AND Pr.description = 'present' AND
    P.description = 'past' AND S.description = 'supine'
    ) ;

-- 5b

CREATE OR REPLACE FUNCTION add_verb() RETURNS trigger AS $$
BEGIN
INSERT INTO Words VALUES(NEW.infinitive, NEW.infinitive, 'verb', 'infinitive') ;
INSERT INTO Words VALUES(NEW.present,    NEW.infinitive, 'verb', 'present') ;
  INSERT INTO Words VALUES(NEW.past,       NEW.infinitive, 'verb', 'past') ;
INSERT INTO Words VALUES(NEW.supine,     NEW.infinitive, 'verb', 'supine') ;
RETURN NEW ;
END;
$$
LANGUAGE plpgsql;

CREATE TRIGGER AddVerb
  INSTEAD OF INSERT ON Verbs
  FOR EACH ROW
  EXECUTE PROCEDURE add_verb() ;
```

6 Indexes

```
-- all analyses of string

SELECT * FROM Words WHERE string = 'lcker' ;
-- 50; with index on string, 1+k where k is the number of different analyses


WITH Forms AS (
```

```
  SELECT *
  FROM Verbs
  WHERE 'lcker' in (infinitive, present, past, supine)
  )
SELECT 'lcker' as string, infinitive as lemma, 'verb' as class,
           'infinitive' as description
FROM Forms
WHERE infinitive = 'lcker'
  UNION
SELECT 'lcker' as string, infinitive as lemma, 'verb' as class,
            'present' as description
FROM Forms
WHERE present = 'lcker'
  UNION
SELECT 'lcker' as string, infinitive as lemma, 'verb' as class,
            'past' as description
FROM Forms
WHERE past = 'lcker'
  UNION
SELECT 'lcker' as string, infinitive as lemma, 'verb' as class,
            'supine' as description
FROM Forms
WHERE supine = 'lcker'
;
-- UNION the same from nouns and adjectives
-- cost 30 = 10+10+10; with indexes on each form, vf + nf + af,
--   certainly larger than in lookup from Words


-- all forms of lemma+class

SELECT * FROM Verbs WHERE infinitive = 'lcka' ;
-- or Nouns or Adjectives
-- 10; with index on lemma, 2 = 1+k where k=1 by assumption


SELECT description, string
FROM Words
WHERE lemma='lcka'
  AND class='verb' AND description IN ('infinitive','present','past','supine') ;
-- 50; with index on (lemma,class), 2 = 1+k where k=1 by assumption

/*
description | string
------------+--------
 infinitive  | lcka
```

```
 past        | lckte
 present     | lcker
 supine      | lckt

which is good enough for the purpose.

If you want

 infinitive | present |  past  | supine
------------+---------+--------+--------
 lcka       | lcker   | lckte | lckt

expand it to the following, with the same cost:
*/

WITH Forms AS (
  SELECT description, string
  FROM Words
  WHERE lemma='lcka' AND class='verb' AND
    description IN ('infinitive','present','past','supine')
  ),
  Infinitives AS (SELECT string AS infinitive FROM Forms WHERE description = 'infinitive'),
  Presents    AS (SELECT string AS present    FROM Forms WHERE description = 'present'),
  Pasts       AS (SELECT string AS past       FROM Forms WHERE description = 'past'),
  Supines     AS (SELECT string AS supine     FROM Forms WHERE description = 'supine')
SELECT * FROM Infinitives, Presents, Pasts, Supines
;
```