Extraction of Fact Tables from a Relational Database: An Effort to Establish Rules in Denormalization

Luís Cavique[1(✉)], Mariana Cavique[2], and António Gonçalves[3]

[1] Universidade Aberta, MAS-BioISI, 1269-01 Lisbon, Portugal
Luis.Cavique@uab.pt
[2] Universidade Europeia, 1500-210 Lisbon, Portugal
MarianaCavique@hotmail.com
[3] Escola Superior de Tecnologia de Setúbal, Instituto Politécnico de Setúbal, 2914-504 Setúbal, Portugal
Antonio.Goncalves@estsetubal.ips.pt

Abstract. Relational databases are supported by very well established models. However, some neglected problems can occur with the join operator: semantic mistakes caused by the multiple access path problem and faults when connection traps arise. In this paper we intend to identify and overcome those problems and to establish rules for relational data denormalization. Two denormalization forms are proposed and a case study is presented.

Keywords: Fact tables · Join operator · Multiple access path problem · Fan trap · Denormalization forms

1 Introduction

The normalization forms in relational databases are a well-known subject with references in many research and teaching documents. On the other hand, the denormalization in relational databases is usually seen as the application of the join operator over a set of tables/relations Ri, R1 |><| R2 |><| … Rn.

Given its simplicity, the relational model and SQL query language have been well accepted and are currently widespread in business environments. However, in some specific database schemas, the result of SQL queries does not match the expected answers. The powerful join operator can lead to undesired situations. The most common are queries with multiple paths between tables return different results (Wald and Sorenson 1984) and some relational inferences can fall into connection traps (Feng and Crowe 1999). Generally join operator problems have been neglected in the literature. The identification of SQL traps has received increasing attention in some professional literature (Business Objects 2007).

In this work we use the concept of fact table given by Kimball and Ross (2013) regarding data warehousing. Data warehouse (DW) provides a stable environment to run complex queries contrary to the use of transactional databases. So, data warehouse design is essential for sustainable Business Intelligence (BI). DW is the basis for the

Á. Rocha et al. (Eds.): WorldCIST'19 2019, AISC 930, pp. 936–945, 2019.
https://doi.org/10.1007/978-3-030-16181-1_88

creation of OLAP (online analytical processing) cubes and multi-dimensional analytical. In typical data warehousing, the ETL (extract, transform, load) process extracts data from operational systems to load it into the data warehouse. Beyond DW, OLAP and BI the migration of relational data to geographic information systems (Kingdon et al. 2016), object oriented tools or web environments (Karnitis and Arnicans 2015) is a relevant issue.

In this work we use the following nomenclature for the tables: lookup tables for table with only cardinality equal to one, intermediate tables for tables with cardinality 1 and N, and fact tables for tables with cardinality N, as shown in Fig. 1. We also draw all tables with relation 1:N this way: the table with a single line is drawn on top, while the table with multiple lines is drawn underneath.

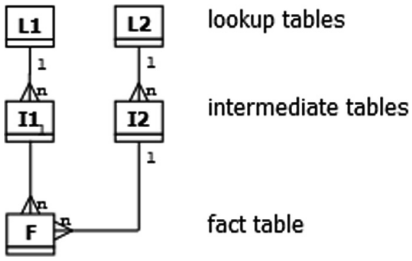


Fig. 1. lookup, intermediate and fact tables

The goal of this paper is to make a review of some neglected join operator problems and to establish two denormalization forms in order to extract fact tables from a relational database. In this work we intend using in the fact tables the same data granularity of the original relational database.

As already referred, database normalization is a well-studied subject, but the bibliography for denormalization is scarce, for this reason a related work section is not presented.

The paper is organized in 5 sections. In Sect. 2 two neglected issues in relational models are recalled: the multiple access path problem and the connection traps. In the same section poly-tree structure is presented. Section 3 presents an effort to establish two denormalization forms taking into account the previous issues. A case study is presented in Sect. 4. Finally, in Sect. 5, some conclusions are drawn.

2 Background Information

Relational model relies extensively on the join operator. However, some difficult situations can occur. This section highlights the query ambiguity, produced by the multiple access paths, which can return queries with different results, and also highlights the connection traps, in particular the fan trap, that return incorrect results.

This section presents two neglected problems with the join operator that affect the extraction of fact tables.

2.1 Multiple Access Path Problem

The Multiple Access Path Problem (MAPP) is presented in (Wald and Sorenson 1984) seeking to translate a sentence into an unambiguous database query. A detailed introduction can be found in (Wu et al. 1996). This problem does not occur in a database where there is a single path between two tables. When two or more paths occur between two tables, the corresponding queries can return different solutions.

In order to systematize the analysis of multiple-path in a schema, it is important to define and classify the paths.

Multiple-paths definition: let the database DB = $\{T_1, T_2, \ldots, T_n\}$ with T_n tables with a set of attributes. A multiple-path occurs if there is more than one path, using the foreign keys, between T_i and T_j tables.

To illustrate the Multiple Access Path Problem, a database of scientific evaluation will be used. Throughout this paper it will be used the database layout considered in Fig. 2. This database schema presents tables with relation 1:N, in which the table with a single line is drawn on top, while the table with multiple lines is drawn underneath.

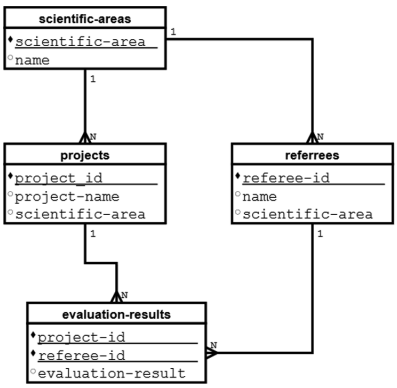


Fig. 2. Database where the Multiple Access Path Problem (MAPP) can occur

In this paper we will use relational algebra symbols: the symbol σ for selection operator, the symbol Π for projection operator, the symbol |><| for the join operator and the symbol $G_{function}$ to identify the aggregation functions.

Choosing tables Scientific-areas and Evaluation-results, two different queries can be defined to obtain the number of evaluations by scientific area. The queries with different paths are:

Q1: G_{count} ($\Pi_{scientific\text{-}area}$ (Scientific-area |><| Projects |><| Evaluation-results))
Q2: G_{count} ($\Pi_{scientific\text{-}area}$ (Scientific-area |><| Referees |><| Evaluation-results))

Using the following data:

Scientific-areas = ((A,_), (B,_), (C,_))
Projects = ((P1,_,A), (P2,_,A), (P3,_,B))
Referres = ((R1,_,C), (R2,_,C), (R3,_C))
Evaluation-results = ((P1, R1,_), (P2, R2,_), (P3,R3,_))

The queries would return different values, Q1 = ((A,2), (B,1)) and Q2 = ((C,3)), showing that queries using different paths between two tables can return different outputs. We present poly-tree structure in order to avoid such inconsistencies.

2.1.1 Poly-tree Structure

An oriented tree is a direct acyclic graph with an in-degree node equal to one, excepting of the root, where the in-degree is equal to zero. A poly-tree is a relaxed oriented tree, with one (and only one) path between any pair of nodes, where the in-degree of a node can be greater than one. Trees and poly-trees have N nodes and N-1 arcs. The poly-tree structure was widespread in Bayesian (or belief) networks (Darwiche 2009).

The first algorithm for Bayesian networks was restricted to trees (Pearl 1982) and was followed by a generalization that became known as the poly-tree algorithm (Pearl 1986). The term poly-tree was coined by (Rebane and Pearl 1987) and the poly-tree algorithm was the very first exact inference algorithm for Bayesian networks.

The goal of a Bayesian network is a complete representation of the joint probability of a set of variables. Inference in a Bayesian network means computing the probability of a query variables set, given an evidence variables set.

The complexity of belief network inference depends on the network structure. In poly-trees (or singly-connected) structures the computational time is linear in the size of the network. When the number of arcs is equal or greater than the number of nodes, the structure is called multiply-connected, and the computational time is exponential in the worst case.

There are three basic methods to solve multiply-connected networks: clustering (or merging), loop-cutset conditioning (or split strategy) and simulation (Russel and Norvig 2003). The clustering method and the loop-cutset conditioning transform the multiply-connected network into a singly-connected network, while the simulation generates a large number of instances that give an approximated solution. In Fig. 3 an original multiply-connected belief network is shown and the poly-trees obtained by using the split method and merge method.

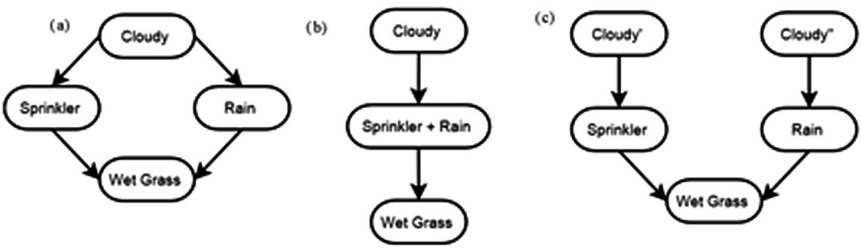


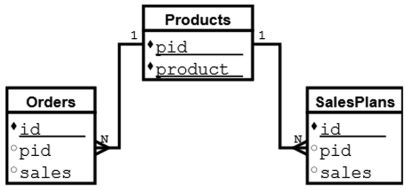
Fig. 3. (a) loopy belief network, (b) merge method, (c) split method

Both opposite strategies of splitting and merging information, using the loop-cutset conditioning/split and clustering/merging methods, contribute to the generation of poly-trees.

2.2 Fan Trap

Connection traps are well-known in database design. A detailed introduction can be found in (Feng and Crowe 1999). The term was coined by Codd when he was proposing the relational model (Codd 1970) and followed by (Date 1995).

To illustrate a connection trap, and in particular the fan trap, a database of ordered and planned sales will be used. The database with a fan schema and a data sample are shown in Fig. 4.



orders		
id	pid	sales
1	1	50
2	1	70
3	2	75
4	2	45
5	3	10
6	3	25

salesplans		
id	pid	sales
1	1	200
2	2	200
3	3	200

Fig. 4. Fan schema and data sample where the connection trap occurs

At the first glance, the SQL query which returns the sum of the ordered sales and planned sales can be written as follows:

```
SELECT P.product, SUM(O.sales) as ordered, SUM(SP.sales) as planned
FROM products P, orders O, salesplans SP
WHERE P.pid = O.pid
AND P.pid = SP.pid
GROUP BY P.product
```

However, the three tables' junction will inflate the value of the planned sales, returning the value of 400 for each product, instead of 200.

In Fig. 4 (on the left), the table Products (on the top) and the tables Orders and SalesPlans (on the bottom) identifies a specific pattern, called in this paper as fan schema. The fan schema in a database is synonymous of connection trap. When a fan schema is found in a database, the data must be aggregated in three steps. For the example shown, first aggregate orders, then aggregate sales and finally join orders and sales.

Connection traps, as referred by (Feng and Crowe 1999), have been defined as 'the lack of understanding of a relational composition' by (Codd 1970), 'false inference' by (Date 1995), 'represent a ternary relation as two binary relations' by (Cardenas 1985) and more explicitly as 'an intrinsic deficiency of the relational theory' by (Ter-Bekke 1992).

3 Denormalization Forms

In this section we propose two denormalization forms. The first denormalization form is when the database has a poly-tree schema, avoiding queries with multiple paths. The second denormalization form is when the poly-tree schema is divided into trees that

forbid fan traps. In the second denormalization form the join operator can be applied over a set of tables/relations Ri, R1 |><| R2 |><| … Rn.

3.1 First Denormalization Form: Poly-tree Structure

The first denormalization form (1DF) is achieved when the database acquires the shape of a poly-tree. The poly-tree database schema avoids the Multiple Access Path Problem (MAPP) already defined.

The split and merge strategies (Russel and Norvig 2003) applied in the belief networks methods (loop-cutset conditioning/split and clustering/merging) are reused for databases, in order to generate poly-trees.

In this work we adopt only the split strategy to create poly-trees. The split technique example is given with the evaluation of scientific projects schema in Fig. 5. By removing the relation of the referee scientific area, there is loss of information. Thus it is preferable to add a new table which distinguishes the scientific areas of projects from the scientific areas of the referees. The removal of multiple-path is obtained by eliminating of a relation and inserting a alias-table.

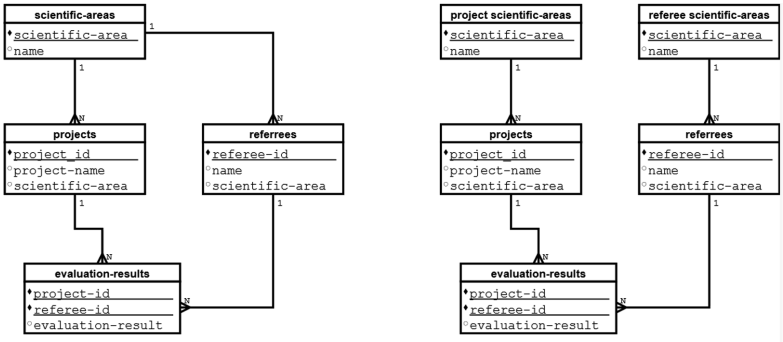


Fig. 5. Split table technique

3.2 Second Denormalization Form: Divide into Trees

The second denormalization form (2DF) is achieved when the database acquires the shape of one or many trees. In a tree database schema, with only one fact table, the fan trap does not occur.

In Fig. 6a a poly-tree, in the first denormalization form, is shown. Tables B and I are fact tables, because they only have relations with cardinality N. If we have more than one fact table the schema is not in the second denormalization form. The join of the tables C |><| D |><| G will cause a fan trap, so the poly-tree should be divided into two trees represented in Fig. 6b.

In the second normal form any join can be performed in each tree, like A |><| B |><| C |><| E |><| F or I |><| J |><| L |><| K |><| H.

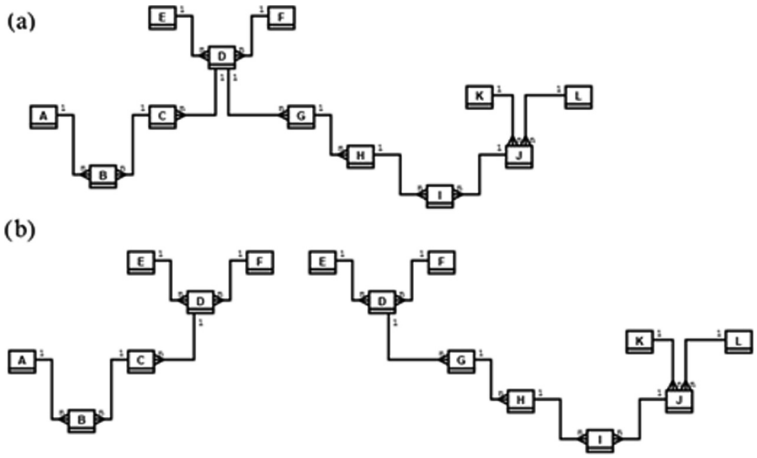


Fig. 6. Database denormalization: (a) poly-tree with two fact tables, (b) two trees

4 Case Study

To illustrate the proposed method, the MP3 database schema, with 16 tables, presented in (Ramos 2006, p. 74) will be used. The database stores information about MP3 playlists with songs from album tracks performed by different artists. To visualize the database, in Fig. 7, we divide the tables into three levels: lookup tables level, intermediate tables level and fact tables level with 8, 3 and 5 tables respectively. In the same figure, multiple-path and fan trap schema can be identified.

To reach the first denormalization form (1DF) a poly-tree database schema must be found. For the MP3 database the following steps were performed:

- remove all lookup tables in order to simplify the visualization; after the denormalization the lookup tables can be included;
- applying the split strategy to remove the multiple-path, by duplicating table Album_1 related to table Track;

Figure 8 presents the MP3 database in the 1DF with a poly-tree structure. In the 1DF the database avoids the semantic ambiguity of multiple paths, but given the number of linked fact tables fan trap can occur.

To reach the second denormalization form (2DF) a tree database schema must be found. For the MP3 poly-tree the following steps were performed:

- apply split technique by duplicating table Artist and isolating the tree (Artist_1, Composition);
- apply split technique by duplicating tables Artist, Album and Track and isolate the tree (Album_2, Track_1, Artist-Track, Artist_2);
- the previous procedures isolated tree (Album, Album-Artist, Artist) and tree (Album_1, Track, PlayList).

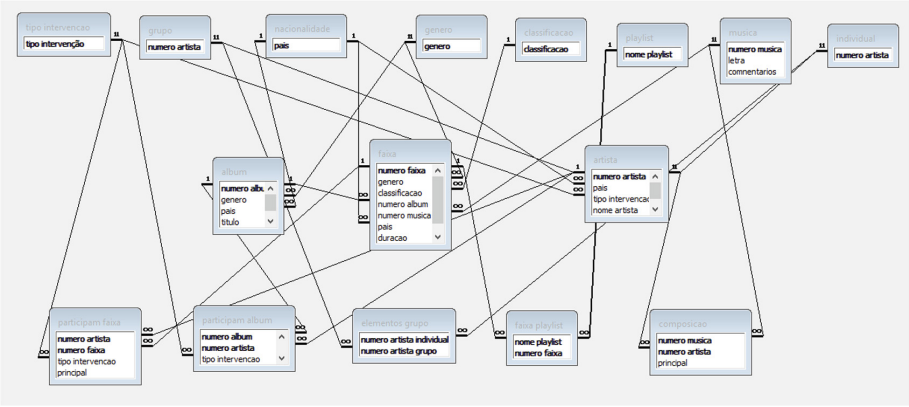


Fig. 7. MP3 database presented in three levels: lookup tables level, intermediate tables level and fact tables level

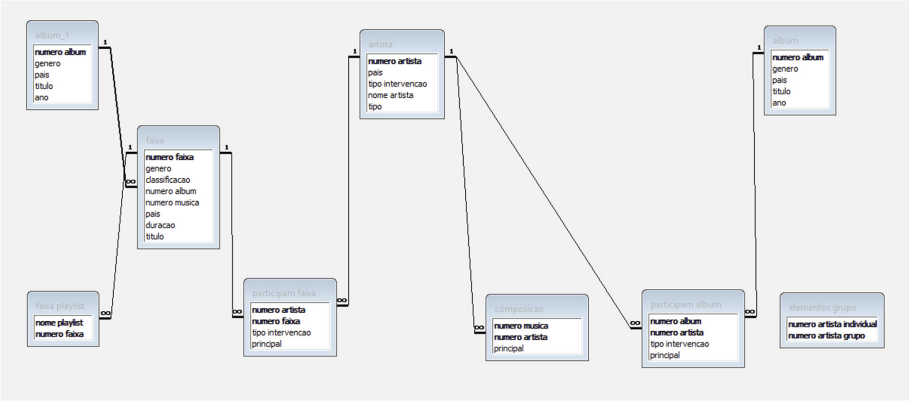


Fig. 8. MP3 database in the 1DF with a poly-tree structure

With the database in the 2DF, all the join operators can be performed without any restriction. Each fact table can be joined with the related tables. Given the 5 fact tables and the 8 lookup tables, a matrix fact tables versus dimensions (or lookup tables) can be drawn in order to give an overview of the database, as it is usual in data warehousing (Fig. 9).

fact tables/ dimensions	1	2	3	4	5
1	X		X		X
2		X		X	
3			X	X	X
4	X	X			
5	X			X	X
6		X	X	X	
7		X			
8			X		X

Fig. 9. Matrix fact table vs dimensions

5 Conclusions

The relational model is one of the most widespread ways to represent data. However, some problems can occur with the join operator: semantic mistakes caused by the multiple access path problems (MAPP) and faults when fan traps arise.

The goal of this paper is to make a review of some join operator problems and to establish two denormalization forms in order to extract fact tables from a relational database.

The first denormalization form (1DF) is achieved when the database acquires the shape of a poly-tree. The poly-tree database schema avoids the multiple access path problem.

The second denormalization form (2DF) is achieved when the database acquires the shape of one or many trees. In a tree database schema, with only one fact table, the fan trap does not occur.

This contribution is an effort to establish formal rules in denormalization and is relevant in ETL (extract, transform, load) for Data Warehousing (DW) and Business Intelligence (BI), but also for geographic information systems (GIS), object oriented tools and web environments and other systems that reuse relational data.

References

Business Objects: Designer's Guide, XI Release 2 (2007)

Cardenas, A.F.: Data Base Management Systems, 2nd edn. Allyn and Bacon, Boston (1985)

Codd, E.F.: A relational model of data for large shared data banks. Commun. ACM **13**(6), 377–387 (1970)

Darwiche, A.: Modeling and Reasoning with Bayesian Networks. Cambridge University Press, New York (2009). ISBN 0521884381

Date, C.J.: Introduction to Database Systems, 6th edn. Addison-Wesley Publishing Company, Reading (1995). ISBN 020154329X

Feng, J., Crowe, M.: The notion of 'classes of a path' in ER schemas. In: Proceedings of Third East European Conference on Advances in Databases and Information Systems, ADBIS 1999, pp. 218–231. Springer, Berlin (1999)

Karnitis G., Arnicans, G.: Migration of relational database to document-oriented database: structure denormalization and data transformation. In: 7th International Conference on Computational Intelligence, Communication Systems and Networks (CICSyN) (2015). https://doi.org/10.1109/cicsyn.2015.30

Kimball, R., Ross, M.: The Data Warehouse Toolkit: The Definitive Guide to Dimensional Modeling, 3rd edn. Wiley and Sons Inc, Hoboken (2013). ISBN 9781118530801

Kingdon, A., Nayembil, M.L., Richardson, A.E., Smith, A.G.: A geodata warehouse: using denormalisation techniques as a tool for delivering spatially enabled integrated geological information to geologists. Comput. Geosci. **96**, 87–97 (2016)

Pearl, J.: Reverend Bayes on inference engines: a distributed hierarchical approach. In: Proceedings American Association of Artificial Intelligence National Conference on AI, Pittsburgh, PA, pp. 133–136 (1982)

Pearl, J.: Fusion, propagation, and structuring in belief networks. Artif. Intell. **29**, 241–288 (1986)

Ramos, P.N.: Desenhar Bases de Dados com UML. Edições Sílabo, Lisbon (2006). ISBN 9789726184744

Rebane, G., Pearl, J.: The recovery of causal poly-trees from statistical data. In: Proceedings of Uncertainty in Artificial Intelligence, pp. 222–228 (1987)

Russell, S.J., Norvig, P.: Artificial Intelligence: A Modern Approach, 2nd edn. Pearson Education, London (2003). ISBN 0137903952

Ter-Bekke, J.H.: Semantic Data Modeling. Prentice Hall, New York (1992). ISBN 0138060509

Wald, J.A., Sorenson, P.G.: Resolving the query inference problem using Steiner trees. ACM Trans. Database Syst. (TODS) **9**(3), 348–368 (1984)

Wu, X., Ichikawa, T., Cercone, N.: Knowledge-Base Assisted Database Retrieval Systems. World Scientific, Singapore (1996). ISBN 978-981-02-1850-8