What is a Database?

second edition

Viji Kumar

Chapter 1 – Explanatory Framework

A database is viewed in the following discussions as an organised collection of data that can be accessed, managed, and updated electronically. It is assumed to be designed to enable the collection and storage of information in a structured and efficient way. The information collected in these databases can then be retrieved and manipulated by digital applications or software, making databases an especially useful tool.

A database consists of tables, i.e., rows and columns, with each column being a specific type of data and each row containing a set of values related to that data. The tables can be linked by relationships, which allow data to be analysed across multiple tables. These linked data are what make databases useful.

Incidentally, it would be prudent to bear in mind that there almost certainly will be data protection legislation in most countries to regulate the collection and use of some types of data, e.g., personal data about individuals. In May 2023 Meta (owner of Facebook) was fined €1.2bn by the EU for privacy violations and ordered to suspend transfers of user data to the US.

Databases provide a way for organisations to access copious amounts of data efficiently, arguably enabling these organisations to make better decisions, improve operations, and deliver better services to their customers. However, the danger of bad actors using the data collected for nefarious purposes, e.g., manipulation of an electorate, is an ever-present threat.

Our changing world, especially changes brought about by human activity, necessitates data having to be processed continuously to enable us to be aware of how our environment is evolving. Collecting this data is vital to help formulate policies for the efficient functioning of, among other things, the economy, health and social care and other infrastructure elements of an advanced 21st century nation.

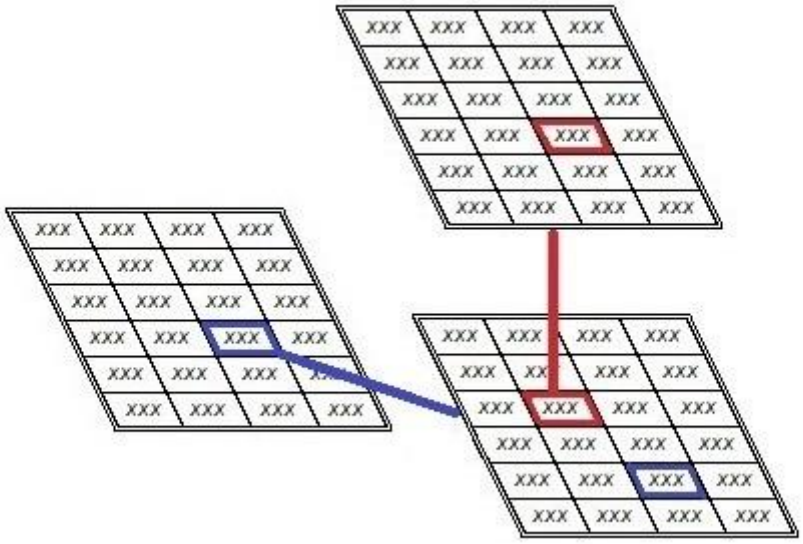
The aim of this book is to enable the reader to review data that they may come across in their daily lives e.g., the results of clinical trials of a vaccine and be able to evaluate the usefulness and the provenance of the data being provided. One of the reasons data such as epidemiological statistics are referenced by any organisation or authority is to justify policies that they are advocating or implementing. Knowing a bit about structured information may be helpful when reviewing the data underpinning such policy justifications. Well-constructed databases allow users to search substantial amounts of data in an efficient fashion to obtain useful insights.

This book is for non-database specialists e.g., students who wish to appreciate some of the factors that must be considered when searching and analysing information presented as tables of data or the visual representations of such data e.g., graphs and charts. Additionally, there will be comments about whether there is any need to probe into the provenance of the data being analysed or reviewed.

team name
falcons
eagles
dolphins
sharks

A list of football teams.

Data are defined as discrete items of information e.g., names of individuals or organisations, dates when events occurred, performance measurements and other metrics. The simplest collection of data is a list e.g., of names of individuals or products. These data items can be linked to other discrete items of information, e.g., addresses, or amounts invoiced producing collections of linked data or databases. The ***function of a database,*** for the purposes of this explanation, is to enable the collection of searchable structured data about related entities e.g., sporting teams and their performances over time in different competitions or musicians and their albums and/or songs.



Here, the ***form of a database*** is viewed as a collection of linked tables. Much of the value of a database arises from the relationships that are implemented by the linkages between its component tables. The rows of these tables are populated by strings of characters. Characters may be letters *(of any alphabet)*, numbers *(i.e., Arabic numerals and zero)*, spaces, punctuation marks and other symbols. The value of each ***data item*** collected and recorded in a database table is an ***attribute*** of an ***entity*** that is of interest to the data collection exercise. e.g., the passport number of an individual crossing a border.

Entities are defined, in this explanation, as constructs that are distinct and with a separate existence e.g., an individual, a credit card or a football team. Entities can also refer to separate, definable, and distinct constructs that have no material existence e.g., league seasons, families, or codons. Examples of data items, i.e., the attributes of diverse types of entities, include an individual's or a team’s name, the time or place of a transaction conducted using a debit or credit card, the venue of a

game of football or the RNA bases in a codon. Some attributes, e.g., unique identifiers *(IDs)* and names, are common to many distinct types of entities, e.g., individuals, objects, and organizations.

Names, identifiers, events, and measurements are “raw data” often collected to enable other information to be derived for the benefit of the users of a database, e.g., the total number of games won by a team during a league season. To enable the collected data to be effectively processed, it is necessary to classify the data items collected. The ***datatype*** is the categorisation used to classify data items in the tables. Data management systems require this categorisation to be able to perform the appropriate operations on the data, as a data item is classified according to its contents e.g., whether the value recorded is text or a number, a number with a decimal component or an integer, or if the data are of specific or esoteric formats such as dates or digital links. The three datatypes featured in this book will be extremely mundane, text strings (e.g., for recording names and unique identifiers), integers (e.g., for recording sequences and the totals of countable entities *(parliamentary seats in the House of Commons)*, and real numbers (e.g., for recording numerical values to two places of decimal).

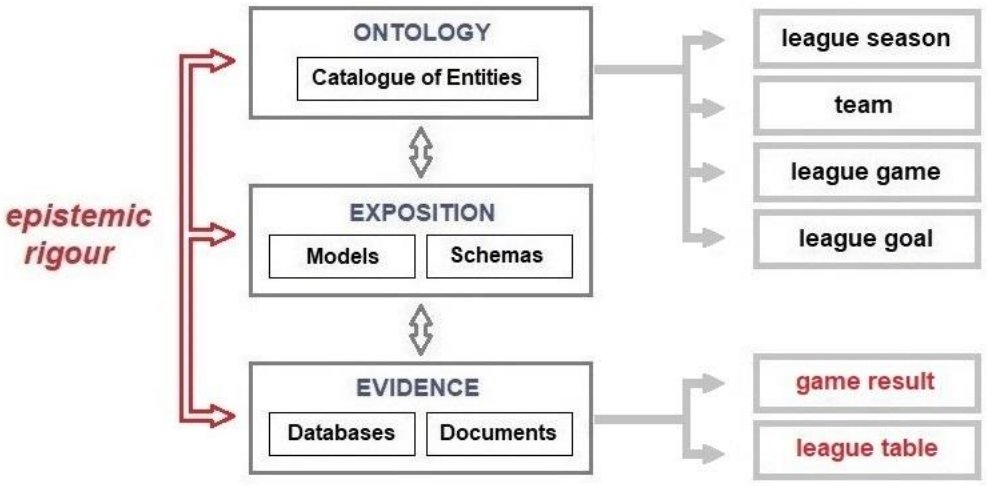
The list of names of football teams shown earlier is of entities that are usually known and referenced by their names. The trouble with names is that they can be changed, e.g., when there is a change of ownership of a company or team. This proclivity of humans to change names will be addressed at the right time because it is often necessary to reference information associated with an entity’s previous name. The ability to change an entity’s name does place an obligation on the custodians of certain types of databases to maintain records of previous names e.g., for a due diligence exercise, it may be useful to look up a company’s assets or liabilities when it was operating under a previous name.

An important feature of the databases discussed here is that the attributes of the entities may be designated as one of three types of ***keys***. This method of structuring and linking data is the invention of Ted Codd and Chris Date, pioneers of database technology in the 1960s. Keys are specifications used by some commonly available digital data management systems e.g., Microsoft Access, to facilitate the provision of structured information. Not to put too fine a point on it, these keys are the crux of the biscuit.

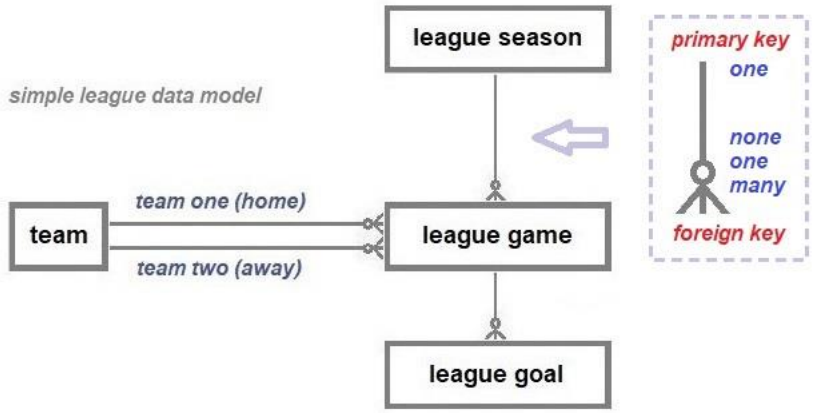
Keys are critical for the linking of the different tables that make up most databases and for the implementation of rules regarding the data being collected. It is horrifying to contemplate the chaos that will be unleashed if keys are not specified to set rules for a data collection exercise. However, keys are optional, and it is possible to structure and use data by other means and techniques e.g., customised documents such as journals and ledgers for financial transactions and, in the natural world, transfer RNA for genetic databases.

Chapter 2 – Designing a Database

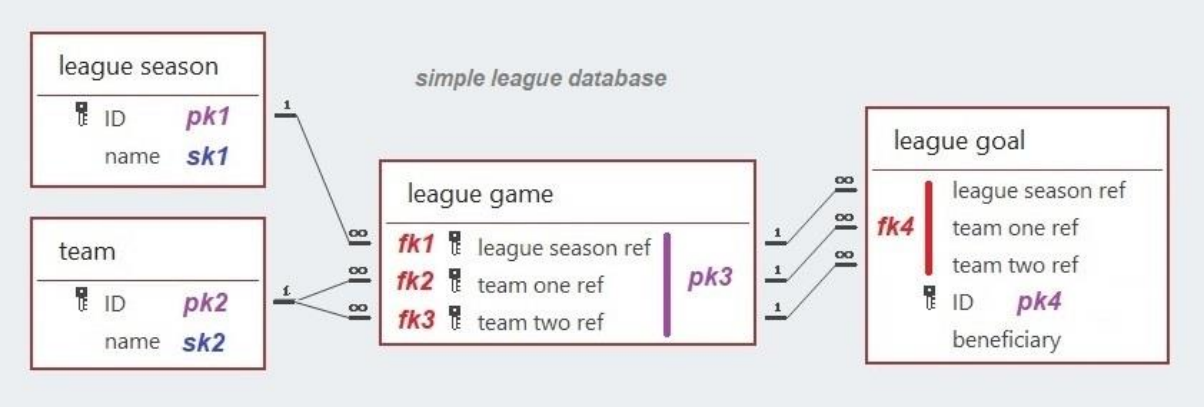
The following constructs will be used to specify the structure of a database; entities, attributes, data items, datatypes, and three types of keys. The three types of keys are ***primary***, ***secondary***, and ***foreign*** keys. The design process will outline how these constructs are deployed to build a data model *(the diagrammatic representation of the linked tables)*, and the utility of the resulting database will be assessed using sample data. This book will fail to meet one of its objectives if, at the end of the book, it is unclear why the databases discussed are structured as they are. This first example will discuss the design of a database to collect data about the games played and goals scored by teams competing over the course of a league season.



To illustrate the construction of a database, four data tables will be linked to enable information to be gleaned about the outcome of round robin football *(soccer – as in As**soc**iation Football)* games, a world-wide phenomenon and one that, I hope, is explicable to most adults *(young and old)*. In this example, teams play each other twice, at home *(team one)* and away *(team two)*. The record of the goals scored in each game is all that will be used to deduce the ranking of the teams at the conclusion of the season, e.g., the model does not allow for the removal of points from a team in the event of a breach of competition rules. The process of designing a database is iterative and may require revisiting and amending the data model as the testing of the integrity and usefulness of the database progresses.



The data model above is a minimalist model constructed to elucidate the nature of the links between the data tables and not to provide a model to satisfy the requirements of afficionados who wish to comprehensively document football league games e.g., the players, the substitutes, and the scorers. However, the model is extensible and can be extended to meet such additional requirements. Primary and foreign keys are the constructs necessary to link the tables.



A ***primary key (pk),*** whether it is a single attribute or a combination of attributes, must be unique, permanent, and unchangeable. The primary key must enable an entity to be uniquely identified when that key is referenced e.g., the combination of an individual's nationality and passport number can be used to identify that individual accurately because it will be unique. To establish a link, an entity records a reference *(the foreign key)* to another entity's primary key. There can only be a single primary key, i.e., only a single attribute or a single set of attributes can be designated as the primary key of an entity.

A ***secondary key (sk)*** is also a designated unique attribute or combination of attributes, but this key can be changed to a different unique value or combination of values. Unlike primary keys, an entity can have more than one secondary key. Secondary keys are used to implement rules, e.g., all teams in a football database must have a unique name as the users are more likely to use names to search the database as opposed to an assigned unique identifier, usually a meaningless character string.

A ***foreign key (fk)*** is the construct used to link the tables of data about the different entity types. An entity can have multiple foreign keys. The link between any two entity types is represented by a line

starting at entity type 1 and ending with a nought and a crow's foot *(or an infinity symbol)* at entity type 2. This type of link, a ***one-to-many***, specifies that there may be none, one or many type 2 entities associated with a single type 1 entity in a specific role and that the type 2 entities must be linked to one and only one type 1 entity,

The functions associated with the use of these keys are the only first principles needed to understand the rest of this explanation, especially the one-to-many relationship that is crucial to define the links between the tables and allow the aggregation of data items as required e.g., counting the total number of goals scored by a team over the course of a football game. The data model above is also specified as a machine-readable schema using a markup language, XML. That XML schema, discussed in chapter 5, can be safely ignored if the data model makes sense. The XML schema is additionally provided in the same helpful spirit that the Rosetta Stone was used to publish a decree about Ptolemy V's reign in three different scripts *(Egyptian hieroglyphs, Demotic and Ancient Greek)*. That thoughtful act helped both, Thomas Young and Jean-Francois Champollion decipher Egyptian hieroglyphs.

The provenance of data collected about sporting competitions can be verified by having been a spectator or by comparing media reports from multiple sources in the public domain. That is very unlike the process of verifying the personal details about individuals even if such data are in the public domain because, unless there is legal access to protected databases, it would be difficult to verify the accuracy of personal data.

The data used to test the *simple league database* in the next chapter are fictitious data. The use of confected data for testing purposes avoids the complications of using data collected by other individuals or commercial entities, data that may require payment for their use and/or be hard to verify. In this case it is the responsibility of the testing team to ensure that the test data covers all possible outcomes of football league games; home wins, away wins and draws including scoreless draws. Note that league football, unlike tournament football, allows draws, and therefore the *simple league* model does not cater for resolving draws e.g., by penalty shoot-outs.

A major responsibility for the custodians of databases is the rigorous testing of data processing software to ensure that it is fit for purpose, i.e., accurately updates a database. In 2022 the courts decided that the malfunctioning of the British Post Office's Horizon computer system between 2000 and 2014 led to a series of events that resulted in great harm to the lives of many sub-postmasters.

Entities must not be multiplied beyond necessity.

In keeping with the imperative expressed by ***Occam's razor***, a principle advocating parsimony, an attempt has been made to keep to a minimum the number of entity types required to meet the objectives of a data collection exercise. Furthermore, it is a truth universally acknowledged that simplicity and extensibility are a database designer's best friends *(see Gall's Law)*, and consequently simplicity and extensibility are presumed here to be useful design principles. The next stage is the

population of the *simple league database* with test data prior to generating the game results and league table.

Chapter 3 – A Simple Database

Shown below are two of the four tables and their attributes, with the primary and the secondary key annotated in each table. The league seasons and the teams are the named entities the data collection exercise is interested in. The other two entity types, the league games and goals are not referenced using names in this model. Returning to the problem with name changes, a meaningless string of characters has been added to the record of each name, a common construct to associate the name of an entity with a unique, permanent, and **unchangeable** attribute, commonly referred to as the unique identifier or ID*, (see below)*.

primary key 1 *secondary key 1*

ID	*name*
f5$m93	**CDL 2021**
h3%w4m	**EPL 2021/22**
vt1&8w	**CDL 2022**
n45LG!	**EPL 2022/23**

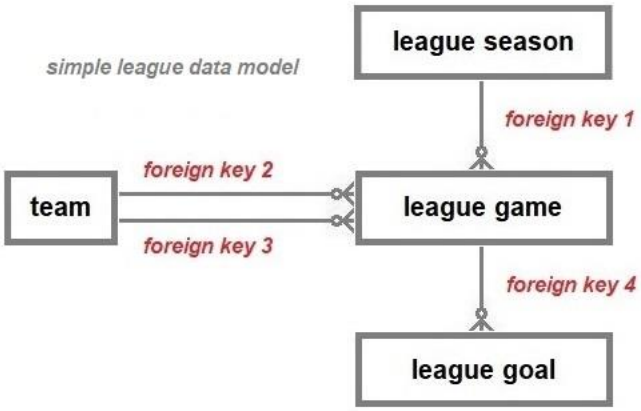
league season

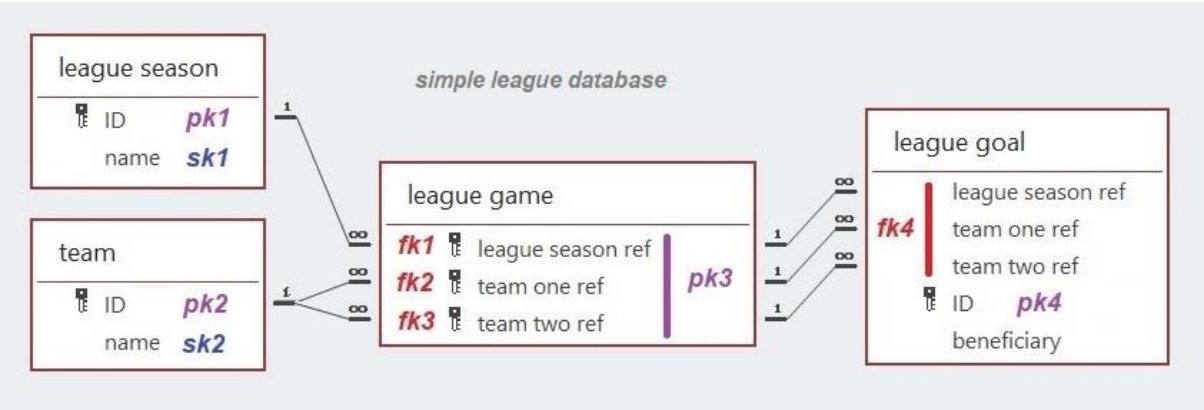
primary key 2 *secondary key 2*

ID	*name*
h6Ap%r	**falcons**
z1c(Y8	**eagles**
e%JiR4	**dolphins**
diT5£j	**sharks**

team

This attribute, the designated primary key, will be used to link to the other tables in the database as required and will allow the name of an entity to be changed while maintaining existing links to other entities. Other attributes about teams and league seasons that are of interest can be added if required. Keys may also be made up of multiple attributes. Furthermore, an attribute can be a part of more than one type of key simultaneously, as shown below.





The model shows an example of a multi-attribute primary key *(pk3)* comprised of three foreign keys *(fk1, fk2, and fk3)* for the *league game* table. That multi-attribute primary key is a multi-attribute foreign key *(fk4)* in the *league goal* table. The league season and the participating teams are linked by the *league game* entity. This is done by recording the primary key of the league season as an attribute of the league games, the *league season ref* attribute and the teams are similarly linked in the role of either the home team *(team one)* or the away team *(team two)*.

foreign key 1 *foreign key 2* *foreign key 3*

primary key 3

league season ref	*team one ref*	*team two ref*
f5$m93	h6Ap%r	z1c(Y8
f5$m93	h6Ap%r	e%JiR4
f5$m93	z1c(Y8	h6Ap%r
f5$m93	z1c(Y8	e%JiR4
f5$m93	e%JiR4	h6Ap%r
f5$m93	e%JiR4	z1c(Y8

league game

[IDs replaced by names]

league season	*team one*	*team two*
CDL 2021	falcons	eagles
CDL 2021	falcons	dolphins
CDL 2021	eagles	falcons
CDL 2021	eagles	dolphins
CDL 2021	dolphins	falcons
CDL 2021	dolphins	eagles

league game

The two tables above contain the same information *(the participating teams of the completed league games)* about the 2021 season of the Charles Darwin League (CDL). The first, the actual database table, displays the unique identifiers and the second is a report that has had the identifiers replaced by the current names of the season and the teams. The fictitious CDL 2021 is a league competition where three teams play each other twice, home, and away, over the course of a season.

The rule that teams in some leagues play each other just twice *(i.e., home, and away)* is used above to define the primary key of a league game. It is a rule that cannot be applied universally to all league seasons, e.g., to the 2022/23 and other recent Scottish Premiership seasons or the Isles of Scilly Football League, where teams play more than a single game against another team either at home and/or away. To accommodate such league competitions, league games will have to be differentiated using alternative or additional attributes.

[IDs replaced by names]

foreign key 4 *primary key 4*

league season	team one	team two	ID	beneficiary
CDL 2021	**dolphins**	**falcons**	**cz%g0F**	**team two**
CDL 2021	**eagles**	**dolphins**	**?e39vL**	**team two**
CDL 2021	**eagles**	**dolphins**	**4Tx2!M**	**team one**
CDL 2021	**eagles**	**dolphins**	**p40a@h**	**team one**
CDL 2021	**eagles**	**falcons**	**2q5H$w**	**team two**
CDL 2021	**eagles**	**falcons**	**L2d5&k**	**team two**
CDL 2021	**falcons**	**dolphins**	**@woL3v**	**team one**
CDL 2021	**falcons**	**eagles**	**b1G&3$**	**team one**
CDL 2021	**falcons**	**eagles**	**k7sl£2**	**team two**

league goal

Assume that there were nine goals scored over the course of the CDL 2021 season and that there was one game that ended goalless. Therefore, that game, the Dolphins *(home)* playing the Eagles *(away)* does not feature in the table of goals above. The only example of a non-key attribute shown is the attribute of greatest interest when reporting results, the *beneficiary* attribute *(i.e., who was the goal awarded to, team one or team two)*.

Each of the twelve types of data items that will populate the four database tables will be one of six distinct types of attributes, *ID*, *name*, *league season ref*, *team one ref*, *team two ref*, and *beneficiary*. The data items are all text strings.

To be of value to football fans, the tables above will need to be extended with data items recording the dates and venues of games, the team sheets for each game, the types of goals scored *(in-play, penalties, and own goals)*, the players who score and other statistics fans may wish to see. The *beneficiary* attribute will be redundant if information is recorded about goal scorers, the type of goal and the team sheets because the beneficiary can then be deduced from that additional information.

Before generating the results of the league games to verify the utility of the *simple league database*, it may be worth revisiting the words of George Box regarding models – “All models are wrong, but some are useful.” An uncompromising view in my opinion but a good reason to ensure that the specification of a model is rigorous as to what it represents, and why any known shortcomings should be clearly highlighted.

Chapter 4 – The Proof is in the Pudding

This example assumes that the league season, CDL 2021, has finished, and all six games have been completed. To produce mid-season league tables, it will be necessary to extend the model and the database to include data about the status of a league game, i.e., whether the game is yet to be played, now in play, or completed.

As you would expect, the *league goal* table is used to obtain the tallies for home and away teams *(by counting the IDs of the relevant goals in each game)*. Here, the two sets of tallies are called *game home tally* and *game away tally*.

game away tally

Field:	league season ref	team one ref	team two ref	tally two: ID	beneficiary
Table:	league goal	league goal	league goal	league goal	league goal
Total:	Group By	Group By	Group By	Count	Group By
Sort:					
Show:	☑	☑	☑	☑	☐
Criteria:					"team two"
or:					

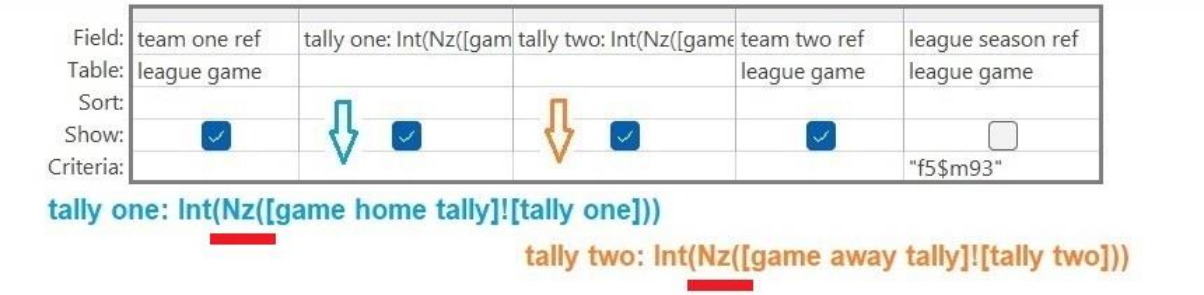
After the counts are completed, the four data items returned are the references to the league season, team one, and team two *(i.e., the three attributes of the primary key of a league game)* together with the total of the goals scored by either the home team or the away team, the total being dependent on the criteria specified for the count *(see above)*. The two tallies obviously make no mention of the teams that fail to score.

league season ref	*team one ref*	*team two ref*	*tally one*
f5$m93	**h6Ap%r**	**e%JiR4**	**1**
f5$m93	**h6Ap%r**	**z1c(Y8**	**1**
f5$m93	**z1c(Y8**	**e%JiR4**	**2**

game home tally

league season ref	*team one ref*	*team two ref*	*tally two*
f5$m93	**e%JiR4**	**h6Ap%r**	**1**
f5$m93	**h6Ap%r**	**z1c(Y8**	**1**
f5$m93	**z1c(Y8**	**e%JiR4**	**1**
f5$m93	**z1c(Y8**	**h6Ap%r**	**2**

game away tally



To generate the results of games, a zero will need to be added as the tally of the non-scoring teams. Therefore, a function will need to be applied when combining the tallies so that a zero is returned when a team is goalless *(e.g., Microsoft Access's null zero function, Nz())*.

Charles Darwin League 2021

team one	*tally one*	*tally two*	*team two*
falcons	**1**	**1**	**eagles**
falcons	**1**	**0**	**dolphins**
eagles	**0**	**2**	**falcons**
eagles	**2**	**1**	**dolphins**
dolphins	**0**	**1**	**falcons**
dolphins	**0**	**0**	**eagles**

The English Premier League (EPL) with twenty teams has 380 games and probably over a thousand goals per season, hence why a fictitious league, with three teams, six games and nine goals, was used to illustrate the construction of a league database. The generation of league tables will not be discussed here as such calculations are well understood, at least since 1888 in England and probably earlier in Scotland.

The league table for the CDL 2021 season shown below is assembled by combining the home and away tables. Three points were awarded for a victory and a single point for a draw.

team one	pld1	win1	drw1	los1	pts1	GD1	for1	agn1
falcons	**2**	**1**	**1**	**0**	**4**	**1**	**2**	**1**
eagles	**2**	**1**	**0**	**1**	**3**	**-1**	**2**	**3**
dolphins	**2**	**0**	**1**	**1**	**1**	**-1**	**0**	**1**

team two	pld2	win2	drw2	los2	pts2	GD2	for2	agn2
falcons	**2**	**2**	**0**	**0**	**6**	**3**	**3**	**0**
eagles	**2**	**0**	**2**	**0**	**2**	**0**	**1**	**1**
dolphins	**2**	**0**	**0**	**2**	**0**	**-2**	**1**	**3**

CDL 2021 League Table

team	pld	win	drw	los	pts	GD	for	agn
falcons	**4**	**3**	**1**	**0**	**10**	**4**	**5**	**1**
eagles	**4**	**1**	**2**	**1**	**5**	**-1**	**3**	**4**
dolphins	**4**	**0**	**1**	**3**	**1**	**-3**	**1**	**4**

Chapter 5 – The Equivalent XML Schema

You can skip this chapter if the preceding four made sense. This chapter will provide the XML *(eXtensible Markup Language)* schema necessary to ensure that the league data in an XML document conform to the *simple league* model introduced earlier. A namespace *(an XML construct)* is required to designate those XML elements that represent the entities in the *simple league* model. The namespace used is ***sls*** *(simple league schema)*. For information about the use of XML, go to the ***World Wide Web Consortium***'s site at www.w3.org/standards/xml/, as there are appropriate tools and languages for providing data as web resources.

The data about the game between the Falcons and the Dolphins that were used in chapter 3 have been provided in an XML document. The document can have any number of the four different entity types as required. The outline of the XML document shows elements which are equivalent to the records from the database tables and have the same names as the tables.

```
<sls:ontology>
  <!--1 league season -->
  <leagueSeason>
     <ID>f5$m93</ID>
     <name>CDL 2021</name>
   </leagueSeason>
  <!--2 teams -->
  <team>
     <ID>h6Ap%r</ID>
     <name>falcons</name>
   </team>
  <team>
     <ID>e%JiR4</ID>
     <name>dolphins</name>
   </team>
  <!--1 league game -->
  <leagueGame>
     <leagueSeasonRef>f5$m93</leagueSeasonRef>
     <teamOneRef>h6Ap%r</teamOneRef>
     <teamTwoRef>e%JiR4</teamTwoRef>
   </leagueGame>
  <!--1 league goal -->
  <leagueGoal>
     <leagueSeasonRef>f5$m93</leagueSeasonRef>
     <teamOneRef>h6Ap%r</teamOneRef>
     <teamTwoRef>e%JiR4</teamTwoRef>
     <ID>@woL3v</ID>
     <beneficiary>team one</beneficiary>
   </leagueGoal>
 </sls:ontology>
```

The above XML document was validated against the schema discussed below. The schema was validated using the CoreFiling XML Schema Validator. The schema that follows is equivalent to the model constructed in chapter 3.

The XML string datatype, ***xs:string***, is the only datatype used in the schema, i.e., the data items are all text strings. The string datatype is further constrained to obtain the specialised simple type that specifies the two options available when recording the *beneficiary* attribute of *league goals*.

There are three sections in this schema: simple types, complex types, and the root element. The simple types are used to construct other simple types and the complex types. The specialised simple type, *beneficiary option*, is used in the specification of the complex type, *league goal defn*. The complex types are used to construct other complex types and the root element. For the *simple league*

database, the root element contains the four entity types that will constitute the database. The complex type *named entity* is used to specify both, the *league season* and the *team* entity types in the root element. The complex type *league goal defn* is an extension of the complex type *league game defn.* The root element defines the elements that can be included in the XML document shown above.

The root element is made up of four elements that are listed together with their primary keys *(4) [aka key]*, secondary keys *(2) [aka unique]* and foreign keys *(4) [aka keyref]*. The four elements and the keys reflect the model described in chapter 3 i.e., the root element is the data model.