COM 230

Record / Database Normalization

Normalization is the process of designing a database in such a way that it is efficient, flexible and free of redundancy. Each step in the process of normalization is termed a from as 1st Normal Form, 2nd Normal Form and so on. Although different businesses use varying levels of normalization, most use a system that involves 1st,2nd and 3rd normal forms

After the identification of the entities and their associated attributes, the design of a relational database can begin with the design of an all inclusive single record. The standard notation is:

recordName(field1, field2, field3…)

Any field that uniquely identifies an entity is called a primary key field and is underlined. This initial record design is termed an un-normalized design.

For a sales application the initial design might look like:

ORDER(ORDER-NUM, ORDER-DATE, PRODUCT-NUM, PRODUCT-NAME, NUM-ORD, UNIT-PRICE)

A set of records might be envisioned in a format such as:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Rec# | Order-Number | ORDER-DATE | PRODUCT-NUM | PRODUCT-NAME | NUM-ORD | UNIT-  PRICE |
| 1 | 00001 | 10/3/2010 | 204 | Widget | 4 | 2.97 |
| 2 | 00002 | 10/3/2010 | 406  204 | Thingie  Widget | 3  10 | 14.99  2.97 |
| 3 | 00003 | 10/3/2010 | 103  406  204 | Dodad  Thingie  Widget | 16  10  5 | 15.49  14.99  2.97 |
| 4 | 00004 | 10/3/2010 | 103  204 | Dodad  Widget | 4  75 | 2.97  2.97 |

You will notice that some of the data is not dependent on the presumed key field. The value for the product name, unit price and the number ordered are not dependent on the order number. These fields are dependent on the product number. For each order there are from one to several products ordered. The PRODUCT-NUM, PRODUCT-NAME, NUM-ORDERED and UNIT-PRICE are called a repeating group. To clarify our initial design we will segregate the repeating group and designate a key for that group, PRODUCT-NUM.

ORDER(ORDER-NUM, ORDER-DATE,( PRODUCT-NUM, PRODUCT-NAME, NUM-ORD, UNIT-PRICE))

The fields that are dependent on PRODUCT-NUM will change with each change in the product number. These fields are said to be functionally dependent on PRODUCT-NUM as a change in the product number will change the value in at least the PRODUCT-NAME AND UNIT-PRICE FIELDS.

First Normal Form:

A record is said to be in First Normal Form if there are no repeating groups. To make the conversion we expand the key to include the key for the repeating group. The key is now ORDER-NUM PRODUCT-NUM. The key is now termed a compound or composite key.

ORDER(ORDER-NUM, ORDER-DATE, PRODUCT-NUM, PRODUCT-NAME, NUM-ORD, UNIT-PRICE)

Our table of orders would now look like:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Record# | Order-Number | ORDER-DATE | PRODUCT-NUM | PRODUCT-NAME | NUM-ORD | UNIT-PRICE |
| 1 | 00001 | 10/3/2010 | 204 | Widget | 4 | 2.97 |
| 2 | 00002  00002 | 10/3/2010  10/3/2010 | 406  204 | Thingie  Widget | 3  10 | 14.99  2.97 |
| 3 | 00003  00003  00003 | 10/3/2010  10/3/2010  10/3/2010 | 103  406  204 | Dodad  Thingie  Widget | 16  10  5 | 15.49  14.99  2.97 |
| 4 | 00004  00004 | 10/3/2010  10/3/2010 | 103  204 | Dodad  Widget | 4  75 | 15.49  2.97 |

Second Normal Form:

A record is in Second Normal Form if it is in First Normal Form and all fields are functionally dependent on the entire primary key. We can note that the order date is not functionally dependent on the product number. Nor are the product name and unit price dependent on the order number. To move to Second Normal form we remove each key field to its own record along with all the fields that are functionally dependent on that portion of the primary key. We then create records that include the keys taken 2 by 2, 3 by 3 until all possible combinations have been created. The fields from the original record design are moved to the table in which that field is dependent on the entire key for that table.

The resulting record designs look like:

ORDER

|  |  |  |
| --- | --- | --- |
| Record# | ORDER-NUM | ORDER-DATE |
| 1 | 00001 | 10/3/2010 |
| 2 | 00002 | 10/3/2010 |
| 3 | 00003 | 10/3/2010 |
| 4 | 00004 | 10/3/2010 |

PRODUCT

|  |  |  |  |
| --- | --- | --- | --- |
| Record# | PRODUCT-NUM | PRODUCT-NAME | UNIT-PRICE |
| 1 | 103 | Dodad | 2.97 |
| 2 | 204 | Widget | 14.99 |
| 3 | 406 | Thingie | 15.49 |

ORDER-LINE

|  |  |  |  |
| --- | --- | --- | --- |
| Record# | ORDER-NUM | PRODUCT-NUM | NUM\_ORD |
| 1 | 00001 | 204 | 4 |
| 2 | 00002 | 406 | 3 |
| 3 | 00002 | 204 | 10 |
| 4 | 00003 | 103 | 16 |
| 5 | 00003 | 406 | 10 |
| 6 | 00003 | 204 | 5 |
| 7 | 00004 | 103 | 4 |
| 8 | 00004 | 204 | 75 |

Third Normal Form:

A record design is said to be in Third Normal Form is the record design is in Second Normal Form and no non-key field is functionally dependent on another non-key field. For example:

ORDER

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Record# | ORDER-NUM | ORDER-DATE | REP-NUM | REP-NAME |
| 1 | 00001 | 10/3/2010 | 1402 | Sally Sellsalot |
| 2 | 00002 | 10/3/2010 | 1645 | Fred Finkelstein |
| 3 | 00003 | 10/3/2010 | 1407 | Bill Smith |
| 4 | 00004 | 10/3/2010 | 1706 | Gary Gruesome |

In the expansion of the order table to include the sales representative information two fields were added. The REP-NAME field is not functionally dependent on the primary key for the table. Instead it is instead dependent upon another non-key field, REP-NUM.

There are problems with this design. If we were to hire a new sales rep it would be necessary to create a dummy order to introduce him to the system. On the other hand to eliminate a rep who has been fired it would be necessary to delete all records in which that rep appears.

In order to move to 3rd normal form this new key and its dependent field must be moved to a new table.

REPS

|  |  |  |
| --- | --- | --- |
| Record# | REP-NUM | REP-NAME |
| 1 | 1407 | Bill Smith |
| 2 | 1402 | Sally Sellsalot |
| 3 | 1645 | Fred Finkelstein |
| 4 | 1706 | Gary Gruesome |

Transition to Third Normal Form may be seen as a final sanity check. It has been said, “A record design is in Third Normal Form if all non-key fields are dependent on the primary key, the entire primary key and nothing but the primary key.”