Database Fundamentals Assignment

Christian Oxner

Study.com

SQL Commands:

1: Select all borrowers

```
SELECT *
FROM borrowers;
```

2: Selecting all books borrowed from borrowers and sorting by order date
Solving the problem this way made the most sense it only selects Books that are present in borrowers. It would also allow us to select fields from both tables to make the data more presentable. The order by clause sorts the query by books that have been checked out the longest. Did not clean up what info to select from each table just selected all info

```
SELECT *
FROM book
JOIN borrower
ON borrower.bookid = book.bookid
ORDER BY borrowdate;
```

3.Used a join statement again to match the author id field in both tables. Selected just the title of the book to display with the authors first and last name as it makes the query look good.

```
SELECT book.booktitle, author.authorfirstname, author.authorlastname
FROM book
JOIN author
ON book.authorid = author.authorid;
```

4. Two different syntaxes one I insert all the data required one I just enter the clientid and then the occupation of pilot

```
INSERT INTO client
VALUES (1, 'Christian', 'Oxner', '1996-07-29', 'Pilot');
```

```
INSERT INTO client (clientID, occupation)
VALUES (1, 'Pilot');
```

Made a sample database to test the queries tried to keep the same naming practices as the assignment provided. All tables were created in lowercase letters.

Database Evaluation Essay

After analyzing the miniature database given for this assignment, I can determine that the database is a relational database (Zandbergen, 2013). The miniature database tables are linked together in various meaningful ways. The best way to begin understanding the structure of this miniature database is by classifying the suppliers and purchases tables as the master tables. From doing this we can determine that the two primary keys in this miniature database are the ITEM_ID and the SUPPLIER_ID (Oglesby, 2016). I can determine that these are the primary keys because they hold a unique value that allows us to find an exact record in the miniature database; while avoiding confusion with similar data.

In the three-remaining produce, grains, and animal products table ITEM_ID and SUPPLIER_ID are foreign keys (Oglesby, n.d.). The cardinality taking place within the database represents a many-to-many relationship (Price, 2017). Knowing that a many-to-many relationship is taking place we can determine that the produce, grains, and animal products tables are all junction tables. They are considered junction tables because they link two of the primary keys together within a single table. One small problem this miniature database presents is that attribute heads are all uppercase instead of the expected lower case.

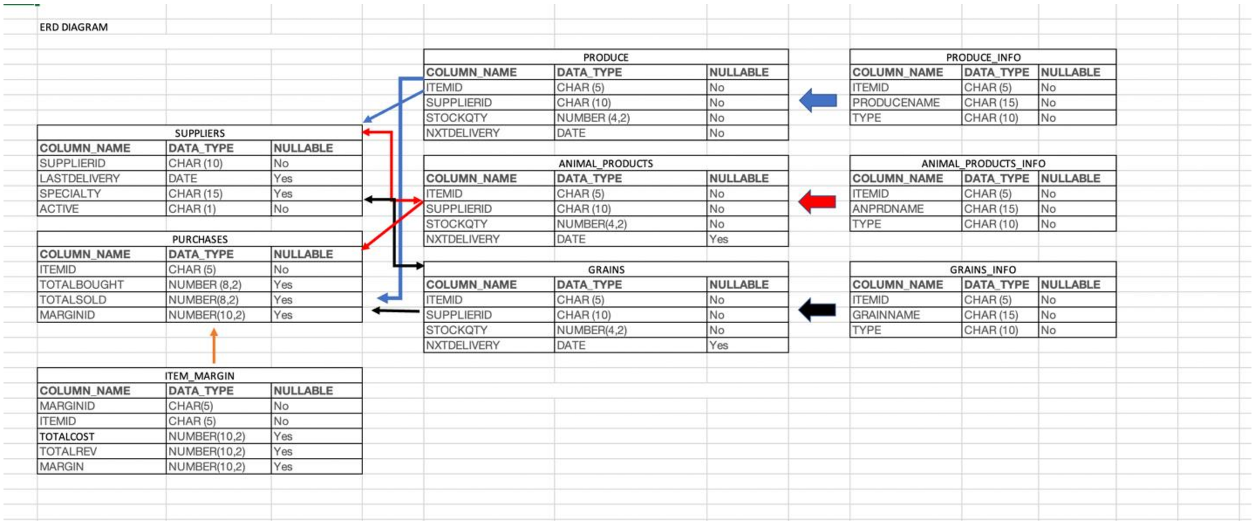
When beginning to look at the normalization of the database we will first check to see if the database is in first normal form. The tables in the database are said to be in first normal form if each column contains single values, columns have unique names, values are of the same data type, no two records are identical (First Normal Form in DBMS with Examples, 2020). The use of ITEMID across the Produce, Animal Products, Grains, and Purchases tables ensures that all records are unique. The same can be said with the SUPPLIERID key in the Suppliers table. Looking at the overall structure of the database it is fair to assume that it is in first normal form.

To begin checking if the database is in second normal form it first has to be in first normal form which we already established it is. The database also has to have no partial dependencies tied to the primary key (Second Normal Form in DBMS with Examples, 2022). The Produce, Animal Products, Grains tables all have partial dependencies to the primary key ITEMID. In the produce table PLUCODE, PRODUCENAME and TYPE can be seen as partial dependencies. In the Animal Products table ANPRDNAME and TYPE can be viewed as partial dependencies. Lastly in the Grains table GRAINNAME and TYPE can be looked at as partial dependencies. The suppliers table has no partial dependencies in it meaning it is in second normal form. The final table Purchases has a MARGIN attribute that we could argue is a partial dependency of TOTAL REV but without a cost column I am also going to assume that this table is in second normal form.

The ERD diagram shows my attempt at moving the database into third normal form. For a database to be in third normal form it first has to be in second normal form, it also only contains fields or columns with no transitive dependencies (Third Normal Form in DBMS with Examples, 2020). To move the database into third normal form I started by removing all transitive dependencies from the Produce, Animal Products and Grains table, by creating their own info tables. The toughest decision I had to make was to try and resolve the dependencies in the purchases table. I did this by creating a new table ITEM_MARGIN table which contained the already declared ITEMID, TOTALREV and MARGIN. I added in a TOTALCOST field as it is used in the margin calculation. In the purchases table I still wanted the MARGIN to appear, but it didn’t make sense for MARGIN to appear twice in both tables as a number. That is why I created the MARGINID as a primary key, which in turn allowed me to link it to the purchases table more sensibly.

ERD DIAGRAM

Made ERD on Microsoft Excel tried to find a way to connect tables through crow's feet but it was not possible.



ERD DIAGRAM

SUPPLIERS

COLUMN_NAME	DATA_TYPE	NULLABLE
SUPPLIERID	CHAR (10)	No
LASTDELIVERY	DATE	Yes
SPECIALTY	CHAR (15)	Yes
ACTIVE	CHAR (1)	No

PURCHASES

COLUMN_NAME	DATA_TYPE	NULLABLE
ITEMID	CHAR (5)	No
TOTALBOUGHT	NUMBER (8,2)	Yes
TOTALSOLD	NUMBER(8,2)	Yes
MARGINID	NUMBER(10,2)	Yes

ITEM_MARGIN

COLUMN_NAME	DATA_TYPE	NULLABLE
MARGINID	CHAR(5)	No
ITEMID	CHAR (5)	No
TOTALCOST	NUMBER(10,2)	Yes
TOTALREV	NUMBER(10,2)	Yes
MARGIN	NUMBER(10,2)	Yes

PRODUCE

COLUMN_NAME	DATA_TYPE	NULLABLE
ITEMID	CHAR (5)	No
SUPPLIERID	CHAR (10)	No
STOCKQTY	NUMBER (4,2)	No
NXTDELIVERY	DATE	No

ANIMAL_PRODUCTS

COLUMN_NAME	DATA_TYPE	NULLABLE
ITEMID	CHAR (5)	No
SUPPLIERID	CHAR (10)	No
STOCKQTY	NUMBER(4,2)	No
NXTDELIVERY	DATE	Yes

GRAINS

COLUMN_NAME	DATA_TYPE	NULLABLE
ITEMID	CHAR (5)	No
SUPPLIERID	CHAR (10)	No
STOCKQTY	NUMBER(4,2)	No
NXTDELIVERY	DATE	Yes

PRODUCE_INFO

COLUMN_NAME	DATA_TYPE	NULLABLE
ITEMID	CHAR (5)	No
PRODUCENAME	CHAR (15)	No
TYPE	CHAR (10)	No

ANIMAL_PRODUCTS_INFO

COLUMN_NAME	DATA_TYPE	NULLABLE
ITEMID	CHAR (5)	No
ANPRDNAME	CHAR (15)	No
TYPE	CHAR (10)	No

GRAINS_INFO

COLUMN_NAME	DATA_TYPE	NULLABLE
ITEMID	CHAR (5)	No
GRAINNAME	CHAR (15)	No
TYPE	CHAR (10)	No

References

Third Normal Form in DBMS with Examples. (2020, March 14). Retrieved from https://study.com/academy/lesson/third-normal-form-in-dbms-with-examples.html.

First Normal Form in DBMS with Examples. (2020, March 16). Retrieved from https://study.com/academy/lesson/first-normal-form-in-dbms-with-examples.html.

Second Normal Form in DBMS with Examples. (2022, May 6). Retrieved from https://study.com/academy/lesson/second-normal-form-in-dbms-with-examples.html.

Oglesby, K. (n.d.). *What is a foreign key?* Study.com. Retrieved from https://study.com/academy/lesson/what-is-a-foreign-key-in-sql.html

Oglesby, K. (2016). *What is a Primary Key in SQL?* Study.com. Retrieved from https://study.com/academy/lesson/what-is-a-primary-key-in-sql.html

Price, R. (2017). *What is the Entity-Relationship Model?* Study.com. Retrieved from https://study.com/academy/lesson/what-is-the-entity-relationship-model-diagram-examples.html

Gloag, D. (2016). *SQL Normalization: Example and Rules*. Study.com. Retrieved from https://study.com/academy/lesson/sql-normalization-example-rules.html

Zandbergen, P. (2013). *What is Relational Database? Elements, Design, Advantages*. Study.com. Retrieved from https://study.com/academy/lesson/what-is-a-relational-database-elements-design-advantages.htm

https://study.com/academy/lesson/second-normal-form-in-dbms-with-examples.html