T3A1.3 - Developer Workbook: Database Systems

Lachlan Greve

Date: 28/11/21

Question 1 :

Describe the properties of a database table in a relational database

A table in a relational database conforms to normalisation rules that are intended to reduce data redundancy and prevent data errors. [^1]

The first normal form (1NF) was introduced by E.F Codd in his publication "A Relational Model of Data for Large Shared Data Banks" (1970) [^2] and described the following properties:

- **Atomicity**: Each data point in a relational table represents a single value. E.g. No table column can contain another table as one of its values.
- **All of the values in a column have the same data type**: This property increases the integrity of the data in the database, preventing the entry of invalid data that wouldn't make sense in the context of the column.
- **Each row is unique**: In a relational database there is no redundancy. Each row will have at least one column that uniquely identifies that particular entry from all others in the database. A unique identifier is often in the form of a primary (or foreign) key and allows for each row of data to be identified and accessed individually.
- **Order of records do not matter**: Relational databases are designed to display, group and manipulate data based on provided specific input queries, therefore the default order of rows or columns is not important.
- **Each column has a unique name***: The name of a column is used to query and select data that it contains. The name of each column needs to be unique otherwise redundant data will be sorted within the database.

In 1971 Codd further defined the second normal form (2NF) Rules[^3]:

- **Table Conforms to 1NF**
- **No Partial Dependencies**: To remove duplicate data from a table, the table should only contain non-key data that is exclusively dependent to the primary key of the table, not just part of it. When the value of a field is not dependent on the primary key of the table, the field should be moved to a separate table and linked to the original table via a reference to the primary key of the original table.

Later in 1971, Codd defined Third Normal Form (3NF) Rules[^3]:

- **Table Must be in 2NF**

- **No Transitive Dependencies**: Similar to 2NF, a table should only contain non-key data that is exclusive to the primary key of the table. A transitive dependency is a field that is dependent on another field, which in turn is dependent to the primary key field. Transitive dependencies should be moved to a child table where the primary key is the column that the columns were functionally dependent on in the original table.

In 1974 Codd and Raymond F. Boyce defined the Boyce-Codd normal form (BCNF)[^4]:

- **Table must be in 3NF**
- **Determinants must be super-keys**: This means that a primary key can not include fields that have a dependency (transitive or partial) on some other field that is not part of the primary key.

An example is a table with fields Student-ID, Course and Educator fields - The Student-ID and Course fields may be combined to form a super-key that can uniquely identify each row. However, since there can be multiple educators for each course, the Course field is dependent on the Educator field and the table does not meet BCNF. To address this two tables must be decomposed into two separate tables with fields including Student/Course and another with Course/Educator.

Question 2 :

Identify the three possible relationships between entities in a relational database and explain how primary and foreign keys are used to create a many-to-many relationship between tables.

The three types of relationships between entities in a relational database are: [^5]

1. One-to-One
2. One-to Many / Many-to-One
3. Many-to-Many

In a many to many relationship the record of one table can be related to one or many records of another and, likewise the records of the second table could be related to one or many records of the first.

In many-to-many relationships a linking table is formed from the primary key property of each table involved in the relationship. Primary keys from other tables referenced in a linking table are known as foreign keys with a new primary key being assigned for the linking table. In this way, a many-to-many relationship is addressed by a relational database by essentially separating the relationship into two one-to-many relationships. [^5]

An example of a many to many relationship is when a customer makes an order from a store. Customers can purchase many items and, products can by purchased by many customers. Therefore, this is a many to many relationship. In this example the two primary keys from each entity (the customer ID and the product ID) would be joined in a linking table named orders. The orders table would be assigned an order_id primary key that is unique to that order and would store both the customer id and product id for the order as foreign keys. In this way, many-to-many relationships can be viewed as two connected one-to-many relationships: a (single) customer can make multiple orders and, multiple orders can contain a product. [^6]

Question 3:

Constraints are restrictions on data. Describe three constrains which can be applied to ensure data integrity in a relational database.

Constraints enforce rules or requirements on data that can be inserted, updated or deleted in a table of a RMDBs. A constraint can be defined as either as a column constraint, which is a condition that applies to data within the column only, or a table constraint - which applies to more than one data column of a table. [^7]

- NOT NULL: This constraint prevent null values from being inserted into a column. A null value has no data type and means that no value has been entered. A nullability constraint is always a column constraint and not a table constraint. A not null constraint can increase data integrity by ensuring that data has been entered into row of a column.
- UNIQUE: The UNIQUE column constraint prevents duplicate values from being inserted into a column. This constraint improves integrity of databases by ensuring that each value in a column is different and preventing duplicate values.
- PRIMARY KEY: A Primary Key constraint combines properties of both NOT NULL and UNIQUE properties. Primary Keys are used to identify each row of a table uniquely and generally has no other significance other than being both UNIQUE and NOT NULL, I.E. - The primary key is used to identify, rather than describe data.

A common syntax used to define constraint is (NOTE: other forms of syntax may also be acceptable):

```
CREATE TABLE table_name(
...,
column_name data_type CONSTRAINT_TYPE
...
)
```

Question 4:

Explain how data types enforce data integrity in a relational database.

Data integrity refers to the accuracy, consistency and reliability of data that is stored in the database. In a relational database, domain constraints are the most elementary way to enforce data integrity. Domain constraints are a user defined set of rules that define the type of data, data length and other attributes that the data must conform to before it can be entered or altered in the database. [^8]

A domain constraint definition consists of two parts:

1. Data type: The data type field specifies the range or domain of values that are valid for an attribute. Relational databases can store data of many different types including: numerical, boolean, date and time types. These values are often specified along with a maximum size or amount of memory that the field can take up in the database, this helps to improve data integrity by preventing invalid data entries that are too large.
2. Constraint: Additional requirements that act as a filter or condition that the data must conform to in order to be entered. Valid constraints for data types include: NOT NULL, UNIQUE, PRIMARY KEY, FOREIGN KEY, CHECK, DEFAULT.

Question 5:

SQL provides operations for the definition and manipulation of data in a relational database. Describe the function of three operators and support your answer with an example.

In SQL, Data Manipulation Language (DML) is defined as language used to for adding, deleting and modifying data in a database. Statements written in DML make changes the data that is stored within database objects but does not make changes to the schema of the database object.

Common DML operations that are included in most Structured Query Language (SQL) distributions include[^9]:

- SELECT: Query data in the database. Allows the database user to retrieve data from the database based on specific input criteria. The `SELECT` statement supports various optional clauses including: The `FROM` clause, which specifies the name of the table(s) to retrieve results from. The `WHERE` clause, which allows filtering of results that meet specific conditions. The `ORDER BY`clause which specified how to order returned rows. The `GROUP BY` clause, which allows for grouping of rows that share a property.
- INSERT: Adds data to a table in a relational database. The values included in the insert statement must satisfy all of the existing constraints that apply to the fields of the table. The `INSERT` statement takes the following required clauses: The `INTO` clause, specifies the table and columns where data is added too. The `VALUES` clause, specifies that data points that are added to the table.
- DELETE: Removes data from a table in a relational database. The `DELETE` command supports the following additional clauses; The `FROM` clause, specifies the table from which data will be deleted from. The `WHERE` clause, which allows filtering of results that meet specific conditions.

Care must be taken to maintain referential integrity when adding, modifying or deleting records with related tables, such as tables that describe many-to-many relationships. For this reason SQL `TRIGGERS` should be added to relational tables to ensure that all necessary changes in response to a command are made at the same time and that the referential integrity of the database is maintained.

The following is an example of a valid SQL syntax query using the `SELECT` command demonstrating the retrieval of OrderTimeKey and SumSales records `FROM` the FactShopSales table[^10]:

```
# Syntax for `SELECT` commands

SELECT OrderTimeKey, SUM(SalesPrice) AS SumSales
FROM FactShopSales
WHERE OrderTimeKey > '20120405'
GROUP BY OrderTimeKey
ORDER BY OrderTimeKey;
```

Question 6:

Relational database make data manipulation highly effective using SQL queries. Describe how the INNER JOIN query retrieves data from multiple tables.

The JOIN query is a type of DML that creates a new tables that returns all rows from tables where the key record of one table is equal to the key values of one or more other tables. An INNER JOIN is the default behaviour of the JOIN query which returns the intersecting records of two or more tables where the specified key value matches. [^11]

The INNER JOIN is differentiated from other types of joins in the following ways:

- FULL OUTER JOIN: Return all values from both tables, displaying NULL values for any columns where records for a key value can't be found.
- RIGHT OUTER JOIN: Returns all records from the right table, displaying NULL values for any columns where records for a key value can't be found in the left table.
- LEFT OUTER JOIN: Returns all records from the left table, displaying NULL values for any columns where records for a key value can't be found in the left table.

Special consideration should be taken when preforming INNER JOINS with tables that contain NULL values. NULL values do not compare equivalently with any other value (including NULL), unless they are specifically addressed with a predicate condition.

An example of valid use of the `INNER JOIN` Syntax is below:

```
SELECT column_name(s)
FROM table1
INNER JOIN table2
ON table1.column_name = table2.column_name
WHERE
```

Bibliography

This syntax compares rows of `table1` with `table2` to check if the condition specified following the `ON` statement is met.

[^1]: Kent, W. (1981) A Simple Guide to Five Normal Forms in Relational Database Theory, Communications of the ACM 26(2).

[^2]: Codd, E. F. (2002). A relational model of data for large shared data banks. In Software pioneers (pp. 263-294). Springer, Berlin, Heidelberg.

[^3]: Codd, E. F. (1972). Further normalization of the data base relational model. Data base systems, 6, 33-64.

[^4]: Codd, E. F., & EF, C. (1975). Recent Investigations in Relational Data Base Systems.

[^5]: Docs.sqlalchemy.org. 2021. Basic Relationship Patterns — SQLAlchemy 1.4 Documentation. [online] Available at: https://docs.sqlalchemy.org/en/14/orm/basic_relationships.html [Accessed 28 November 2021].

[^6]: Docs.oracle.com. 2010. ATG Repository Guide. [online] Available at: https://docs.oracle.com/cd/E26180_01/Platform.94/RepositoryGuide/html/index.html [Accessed 28 November 2021]

[^7]: Grefen, P. W., & Apers, P. M. (1993). Integrity control in relational database systems—an overview. Data & Knowledge Engineering, 10(2), 187-223.

[^8]: Daum, B. (2003). 11 - Reality Check: The World Is Relational. In B. Daum (Ed.), Modeling Business Objects with XML Schema (pp. 395–444). Morgan Kaufmann. https://doi.org/10.1016/B978-155860816-0/50013-6

[^9]: Van Der Lans, Rick F. The SQL standard: a complete guide reference. Prentice Hall International (UK) Ltd., 1989.

[^10]: SELECT (Transact-SQL) - SQL Server. Docs.microsoft.com. (2021). Retrieved 28 November 2021, from https://docs.microsoft.com/en-us/sql/t-sql/queries/select-transact-sql?view=sql-server-ver15.

[^11]: Mahajan, G. (2021). A step-by-step walkthrough of SQL Inner Join. SQL Shack - articles about database auditing, server performance, data recovery, and more. Retrieved 28 November 2021, from https://www.sqlshack.com/a-step-by-step-walkthrough-of-sql-inner-join/.