DATA TECHNOLOGIES
Alex Acquier – ID#: G00293624

Part 4: CODD’S RULES

RULE 1: The information Rule:

“All information in a relational database is represented explicitly at the logical level in exactly one way - by values in tables”.

Queries:

```
SELECT * FROM STUDENTS;                                          Q 1-1
SELECT * FROM ASSESMENT;                                         Q 1-2
SELECT * FROM LECTURERS;                                         Q 1-3
SELECT * FROM COURSES;                                           Q 1-4
```

Queries Q 1-1 through Q 1-4 allows to display all the tables of the database, this show that the database is conform to Codd’s first rule.

DATABASE FOLLOWS RULE 1.

Rule 2: The guaranteed access rule

“Each and every datum (atomic value) in a relational database is guaranteed to be logically accessible by resorting to a combination of table name, primary key value and column name”.

Queries:

```
SELECT Student_Name FROM STUDENTS
WHERE Course_Name= 'BSc. Webtechnologies';                       Q 2-1
SELECT Course_Name FROM COURSES WHERE Course_ID='E_99';           Q 2-2
SELECT Dept_ID FROM LECTURERS WHERE Lecturer_ID=13;              Q 2-3
SELECT Grade FROM ASSESMENT WHERE Student_ID=21;                 Q 2-4
```

Queries Q 2-1 through Q 2-4 show that the database is compliant with Codd’s second rule as each of these queries returns the data of single cell using a column name, a table name and the primary key associated to the table.

DATABASE FOLLOWS RULE 2.

Rule 3: Systematic treatment of null values

"Null values (distinct from the empty character string or a string of blank characters or any other number) are supported in the fully relational DBMS for representing missing information in a systematic way, independent of data type".

Queries:

INSERT INTO COURSES (Course_ID, Course_Name) VALUES ('D59', null); Q 3-1

INSERT INTO STUDENTS (Student_ID, Student_Name, Student_DOB,

Course_Name) VALUES (25, null, '1999-01-01', null); Q 3-2

INSERT INTO ASSESMENT (Ass_ID, Student_ID, Lecturer_ID, Due_Date,

Grade) VALUES ('Website', 25,12,'2016-12-18', null);
Q 3-3

INSERT INTO LECTURERS (Lecturer_ID, Lecturer_Name, Course_ID,

Dept_ID) VALUES (16, null, 'D59', 'Art&Skills'); Q 3-4

The queries Q 3-1 through Q 3-4 all came back with error messages of the type shown below:

```
#1048 - Column 'Lecturer_Name' cannot be null
```

It is because when the tables were created all the column have been set as NOT NULL. This is the result of the business rule that wants that most of the columns should not contain any null values, the only column which could be containing null is the Grade column in the Assesment table. In certain circumstances, a lecturer may dispense a student of an assesment and leave a null value but this has not been considered while the database was designed.

DATABASE DOES NOT FOLLOW RULE 3.

Rule 4: Dynamic online catalog based on the relational model

"The data base description is represented at the logical level in the same way as ordinary data, so that authorised users can apply the same relational language to its interrogation as they apply to regular data".

Query:

SELECT * FROM information_schema.tables Q 4-1

This rule requires that a relational database be self-describing. In other words, the database must contain system tables whose columns describe the structure of the database itself, or alternatively, the database description is contained in user-accessible tables. In other words, this rule insists a data dictionary that stores meta-data.[1]

In MySQL, it is done by accessing the INFORMATION_SCHEMA information database on the left side of the phpMyAdmin window alongside the databases or by using the query Q 4-1 above.

DATABASE FOLLOWS RULE 4.

Rule 5: The comprehensive data sub language rule.

"A relational system may support several languages and various modes of terminal use (for example, fill-in-the-blanks mode). However, there must be at least one language whose statements are expressible, per some well-defined syntax, as character strings and that is comprehensive in supporting all of the following items:

- Data definition.

- View definition.

- Data manipulation (interactive and by program).

- Integrity constraints.

- Transaction boundaries (begin, commit and rollback)".

Queries:

```
CREATE TABLE ASSESMENT (
  Ass_ID varchar (20) NOT NULL,
  Student_ID int (4) NOT NULL,
  Lecturer_ID int (4) NOT NULL,
  Due_Date date NOT NULL,
  Grade varchar (20) NOT NULL,
  PRIMARY KEY (Student_ID)  );                                          Q 5-1

CREATE VIEW lecturerInfo AS SELECT Lecturer_ID, Lecturer_Name,
Dept_ID FROM LECTURERS;                                                 Q 5-2

INSERT INTO STUDENTS VALUES (25, 'Owen Wilson', '1985-05-02',
'BA History');                                                          Q 5-3

ALTER TABLE STUDENTS ADD Ass_ID varchar (20);
ALTER TABLE ASSESMENT
ADD CONSTRAINT Assesment_Lecturers
FOREIGN KEY (Lecturer_ID)
REFERENCES LECTURERS(Lecturer_ID);                                      Q 5-4

UPDATE LECTURERS set Dept_ID='CompSc' WHERE Course_ID='X_55';           Q 5-5
```

The query Q 5-1 demonstrates the data definition part of rule 5 by allowing to create a table and by defining each of the columns of the table as well as their types.
The query Q 5-2 demonstrates the view definition part of rule 5 by allowing to create a view and populating it by importing data from another table.
The query Q 5-3 demonstrates the data manipulation part of rule 5 by inserting new data in an existing table.
The query Q 5-4 demonstrates the integrity constraint part of rule 5 by altering a table to create a foreign key and giving a name to the constraint.
The query Q 5-5 demonstrates the transaction boundaries part of rule 5 by updating cells in a table with a value in condition that another cell in this row contains another value.

MySQL is a Structured Query Language which allows the creation of tables as well as the view and manipulation of the data inside them, it also allows the creation of integrity constraints and to perform transactional processing. I think MySQL follows Codd's fifth rule.

MySQL FOLLOWS RULE 5.

Rule 6: The view updating rule

"All views that are theoretically updateable are also updateable by the system".

Queries:

CREATE VIEW lecturerInfo AS SELECT Lecturer_ID, Lecturer_Name, Dept_ID FROM LECTURERS; Q 6-1

CREATE VIEW AssInfo AS SELECT Ass_ID, Student_ID, Lecturer_ID, Due_Date FROM ASSESMENT Q 6-2

INSERT INTO COURSES VALUES ('H_101','BA History'); Q 6-3

INSERT INTO STUDENTS VALUES (25, 'Owen Wilson', '1985-05-02', 'BA History'); Q 6-4

INSERT INTO LECTURERS VALUES (16,'Dave Duvay', 'H-101', 'Art&Skills'); Q 6-5

INSERT INTO ASSESMENT VALUES ('Brit_Emp.',25, 16, ' 2017-01-03','Need Grading'); Q 6-6

Queries Q 6-1 and Q 6-2 are creating the views necessary for the demonstration of the sixth rule. The queries Q 6-3 through Q 6-6 are updating the tables with new information so the view themselves can be updated but each of them return the error message showed below:

```
#1452 - Cannot add or update a child row: a foreign key constraint fails
```

This is due to the limited number of table and the fact they have very tight constraints, this means that if the table cannot be updated the views are not updatable either and Codd's sixth rule does not apply in this case. If the view would have been updatable, the syntax use would have been:

UPDATE *VIEW_NAME* SET *COLUMN_NAME='DATA_REQUIRED' WHERE COLUMN_NAME=DATA_REQUIRED'*

Delete command would also been attempted using this syntax:
DELETE FROM table_name WHERE some_column = some_value

DATABASE DOES NOT FOLLOW RULE 6.

Rule 7: High level Insert, Update and Delete

"*The capability of handling a base relation or a derived relation as a single operand applies not only to the retrieval of data but also to the insertion, update and deletion of data".*

Queries:

```
INSERT INTO COURSES VALUES ('H_101','BA. History'),
('COOK_18','BA. Fine Cuisine'),
('EL_55','BA. English Literature');                                   Q 7-1

UPDATE LECTURERS set Dept_ID='CompSc' WHERE Course_ID='X_55';          Q 7-2

DELETE FROM LECTURERS WHERE (Dept_ID) IN (('Sc&Comp'), ('CompSc'));    Q 7-3
```

The query Q 7-1 should have inserted multiple values in the courses table but as in the case of Codd’s sixth rule, the error message below appeared;

```
#1452 - Cannot add or update a child row: a foreign key constraint
fails
```

This due to way the database has been designed and this does not comply with Codd’s seventh rule.

The query Q 7-2 updates data already present in the table and yielded positive results and by doing so, complies with Codd’s seventh rule.

The query Q 7-3 deletes data already present in the lecturers table but as it was the case for Q 7-1, this error message appeared:

```
#1451 - Cannot delete or update a parent row: a foreign key
constraint fails
```

This is also due to way the database has been designed and this does not comply with Codd’s seventh rule.

Due to design issue the database has a very limited compliance to Codd’s seventh rule.

DATABASE HAS A VERY LIMITED LEVEL OF COMPLIANCE WITH RULE 7.

Rule 8: Physical data independence

“Applications programs and terminal activities remain logically unimpaired whenever any changes are made in either storage representation or access methods”.

PhpMyAdmin (the DBMS used for this project) offers the possibility to export your database to any other parts of the system or to another server. Firstly, select your database in the left window of phpMyAdmin, after that select the export tab on the right window (5th from the left). Once the tab is opened, give a name to the new template, select your method of export (we will use the quick one), make sure that the file format is SQL and click on the GO button. After that, you have the option of either open the file in a text editor or to save it (save it), you will find the SQL file in your download and you should be able to install it in any system. You can also use an export link to send the data directly to another server under the condition that:

- Both servers have phpMyAdmin
- The database size is under 50MB
- The database file is small enough to overcome upload size limit (can be compressed)

This can also be done through command line and SSH by using the following command:

```
mysqldump  -p  -u  username  database_name > dbname.sql
```

This way also offer the possibility of exporting only one table by using the following command:

```
mysqldump  -p  --user=username  database_name tableName > tableName.sql
```

Both method have been used and the resulting SQL files have been imported into other servers, this mean that the DBMS is complying with Codd's 8th rule.
DBMS COMPLIES WITH RULE 8.

Rule 9: Logical data independence

"Applications programs and terminal activities remain logically unimpaired when Information-Preserving changes of any kind that theoretically permit unimpairment are made to the base tables".

Queries:

```
CREATE TABLE STUDENTS_ADDRESS (
  Student_Name varchar (20) NOT NULL,
  Student_Address varchar (100) NOT NULL,
  City varchar (20) NOT NULL,
  County varchar (20) NOT NULL,
  PRIMARY KEY (Student_Name) );                                    Q 9-1

INSERT INTO STUDENTS_ADDRESS VALUES
('Joe Blogg', '11 O Connell Street', 'Dublin', 'Dublin'),
('Mary Reilly', 'Apt 12a, 52 Eglington Street', 'Galway', 'Galway'),
('Jane Doe', '12 Ocean Drive', 'Dingle', 'Kerry'),
('Dan Smith', 'Apt 25, Bedford Row', 'Kilkenny', 'Kilkenny');     Q 9-2

ALTER TABLE STUDENTS ADD Ass_ID varchar (20);                      Q 9-3

ALTER TABLE STUDENTS ADD CONSTRAINT students_StuAdd FOREIGN
KEY(Student_Name) REFERENCES STUDENTS_ADDRESS(Student_Name)        Q 9-4
```

Queries Q 9-1 through Q 9-3 have been executed without any problem but when query Q 9-4 was introduced, an error message was displayed:

```
#1452 - Cannot add or update a child row: a foreign key constraint
fails
```

The fact that the three first queries could create a table and populate as well as to add a new column into another table is the first indication that the DBMS seems to follow the 9th rule. The fact that query Q 9-4 failed shows that it follows this rule because it shows that the DBMS went to check if all the data meet the constraint and as it did not, returned an error message.

DBMS FOLLOWS RULE 9.

Rule 10: Integrity Independence

"Integrity constraints specific to a particular relational database must be definable in the relational data sublanguage and storable in the catalog, not in the application programs. A minimum of the following two integrity constraints must be supported:

1. *Entity integrity: No component of a primary key is allowed to have a null value.*

***2.** Referential integrity: For each distinct non null foreign key value in a relational database, there must exist a matching primary key value from the same domain.”*

Queries:

1. Entity Integrity

SELECT Course_Name FROM STUDENTS; Q 10-1

SELECT Course_ID FROM COURSES; Q 10-2
SELECT Lecturer_ID FROM LECTURERS; Q 10-3

SELECT Student_ID FROM ASSESMENT Q 10-4

2. Referential integrity

SELECT STUDENTS.Course_Name, COURSES.Course_Name FROM STUDENTS,
COURSES WHERE STUDENTS.Course_Name=COURSES.Course_Name; Q 10-5

SELECT COURSES.Course_ID, LECTURERS.Course_ID FROM COURSES,
LECTURERS WHERE COURSES.Course_ID=LECTURERS.Course_ID; Q 10-6

SELECT LECTURERS.Lecturer_ID, ASSESMENT.Student_ID FROM LECTURERS,
ASSESMENT WHERE LECTURERS.Lecturer_ID=ASSESMENT.Student_ID; Q 10-7

SELECT ASSESMENT.Student_ID, STUDENTS.Student_ID FROM ASSESMENT,
STUDENTS WHERE ASSESMENT.Student_ID= STUDENTS.Student_ID; Q 10-8

The queries Q 10-1 through Q 10-4 allow to display all the data of each of the column referenced as primary key and after observation, there were no entry with a null value which satisfies the first part of rule 10.
The queries Q 10-5 through Q 10-4 allow to display all the matching values of the columns referenced as primary and foreign key. The number of rows returned by those queries matched the number of rows of the table with the greatest number of rows.

Rule 11: Distributed Independence

“The data manipulation sub-language of a relational DBMS must enable application programs and terminal activities to remain logically unimpaired whether and whenever data are physically centralized or distributed”

If the database is split into several parts (i.e.: split in several servers or partitions), this should be invisible to the end user.

Rule 12: Non subversion rule

“*If a relational system has a low-level (single-record-at-a-time) language, that low-level language cannot be used to subvert or bypass the integrity or constraints expressed in the higher level relational language (multiple-records-at-a-time).”*

This means that only a language like SQL should be able to alter the database and no other kind of programming (i.e.: machine language).

REFERENCES:

1 – Name: exploredatabase.com * Title: Codd’s twelve rules – Rule 4 – Dynamic Online Catalog Based on the Relational Model * Link: http://www.exploredatabase.com/2015/02/codds-twelve-rules-rule-4-dynamic-online-catalog-based-on-relational-model.html * Last accessed: 4th of December 2016