Codd's Rules

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

```
SELECT * FROM `customers`;
```

2. Guaranteed Access Rule:

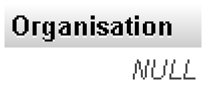
Each and every datum (atomic value) in a relational database is guaranteed to be logically accessible by resorting to a combination of table name, primary key value and column name.

```
SELECT Name FROM `customers` WHERE Customer_id =5
```

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.

```
SELECT Organisation from customers where Name="Mary" AND Surname="Jones"
```



Organisation
NULL

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.

```
SELECT * FROM `INNODB_SYS_TABLES`
```

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:

> **INSERT into customers (Customer_id, Name, Surname, Organisation) VALUES (8, 'Bob', 'Marley', 'The Mechanic');**

6. The view updating rule:

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

> **create view amounts as select Amount_due from orders;**
>
> **update amounts set Amount_due=(Amount_due +10);**
>
> **UPDATE orders set Amount_due = (Amount_due + 20);**

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.

> **UPDATE parts set price= (price + 10.00)**

8. Physical data independence:

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

I changed the name of one of the data files on the physical layer, the file is on the following path: C:\xampp\mysql\data\car_parts and the file was called: car_make.frm and I changed it to car_make1.frm. I then make the following query on that table and it carried out the query as before:

SELECT car_make_id FROM `car_make` WHERE car_make_name='Toyota';

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.

If I make a new table called suppliers it won't affect any of the earlier queries –

CREATE TABLE suppliers (Name varchar(45), Address varchar(45));

INSERT INTO `suppliers` (`Name`, `Address`) VALUES

('AutoYard', 'Galway'),

('Dismantlers', 'Mayo'),

('John's Caryard',' Belfast',

10. Integrity Independence:

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.*

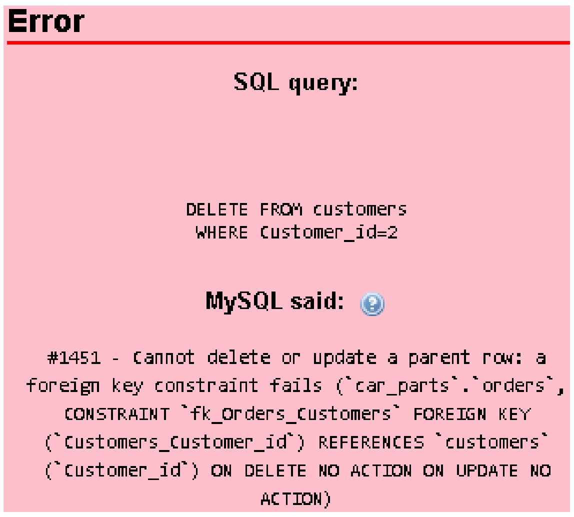
update customers set Customer_id = NULL where Organisation="Smith's";

(Won't let me do this)

***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.*

DELETE FROM customers WHERE Customer_id=2;

This query gives the following error:



Error

SQL query:

```
DELETE FROM customers
WHERE Customer_id=2
```

MySQL said:

```
#1451 - Cannot delete or update a parent row: a foreign key constraint fails (`car_parts`.`orders`, CONSTRAINT `fk_Orders_Customers` FOREIGN KEY (`Customers_Customer_id`) REFERENCES `customers` (`Customer_id`) ON DELETE NO ACTION ON UPDATE NO ACTION)
```