Ollscoil Teicneolaíochta an Atlantaigh
Atlantic Technological University

H. Dip. in Science
Software Development

Codd's Rules

Database Design & Development 2023

Jonathan Griffey – G00299043

Contents

Rule 1: Information rule 2
Rule 2: Guaranteed Access rule 2
Rule 3: Systematic Treatment of Null Values 3
Rule 4: Dynamic Online Catalog 3
Rule 5: Comprehensive Data Sublanguage Rule 4
Rule 6: View Updating Rule 4
Rule 7: High-Level Insert, Update, and Delete 4
Rule 8: Physical Data Independence 5
Rule 9: Logical Data Independence 5
Rule 10: Integrity Independence 5
Rule 11: Distribution Independence 6
Rule 12: Non-Subversion Rule 6
References 6

Rule 1: Information rule

All information in a relational database is represented explicitly at the logical level and in exactly one way – by values in tables. The database created represents all information in a relational format with each table explicitly storing information related to one specific entity, e.g., the Animal table contains information related to the animals treated, and the Staff table contains information related to the staff members in the veterinary practice.

Showing rows 0 - 6 (7 total, Query took 0.0002 seconds.)

```
SELECT * FROM `staff`;
```

	StaffID	firstName	lastName	Role	addressId	contactCode
Edit Copy Delete	stf001	Joe	ODonnell	Vet	add000001	con000001
Edit Copy Delete	stf002	Jonathan	Griffey	Vet	add000002	con000003
Edit Copy Delete	stf003	Tony	Burke	Vet	add000003	con000004
Edit Copy Delete	stf004	Laura	ODonnell	Nurse	add000001	con000005
Edit Copy Delete	stf005	Kate	ODonnell	Nurse	add000001	con000006
Edit Copy Delete	stf006	Michelle	Casey	Nurse	add000004	con000007
Edit Copy Delete	stf007	Roisin	Murphy	Receptionist	add000005	con000002

Rule 2: Guaranteed Access rule

Every single data element (value) is guaranteed to be accessible logically with a combination of table-name, primary-key (row value), and attribute-name (column value). No other means, such as pointers, can be used to access data. For example, the primary key in the staff table is the StaffID, which can be used to uniquely identify each employee.

Showing rows 0 - 2 (3 total, Query took 0.0004 seconds.)

```
SELECT StaffID, firstName, lastName from staff WHERE addressId = 'add000001';
```

	StaffID	firstName	lastName
Edit Copy Delete	stf001	Joe	ODonnell
Edit Copy Delete	stf004	Laura	ODonnell
Edit Copy Delete	stf005	Kate	ODonnell

Rule 3: Systematic Treatment of Null Values

The NULL values in a database must be given a systematic and uniform treatment. This is a very important rule because a NULL can be interpreted as one the following – data is missing, data is not known, or data is not applicable. The database created allows null values in appropriate cases. For example, the contact table allows for null values for emails column, as some pet owners may not have email addresses.

☐	Edit	Copy	Delete	con000009	j.mcgrath@gmail.com	NULL	NULL
☐	Edit	Copy	Delete	con000010	kevin.lynch@gmail.com	+353832791811	NULL
☐	Edit	Copy	Delete	con000011	NULL	+353869874521	NULL
☐	Edit	Copy	Delete	con000012	s.murphy@gmail.com	+353813456999	NULL
☐	Edit	Copy	Delete	con000013	NULL	+353865432198	NULL
☐	Edit	Copy	Delete	con000014	NULL	+353894785632	NULL
☐	Edit	Copy	Delete	con000015	s.casey@gmail.com	+353872000900	NULL
☐	Edit	Copy	Delete	con000016	eoinbarry@gmail.com	+353892221095	NULL

Rule 4: Dynamic Online Catalog

The structure description of the entire database must be stored in an online catalog, known as data dictionary, which can be accessed by authorized users. Users can use the same query language to access the catalog which they use to access the database itself.

Showing rows 0 - ... (Query took 0.4767 seconds.)

```
SELECT * FROM information_schema.tables;
```

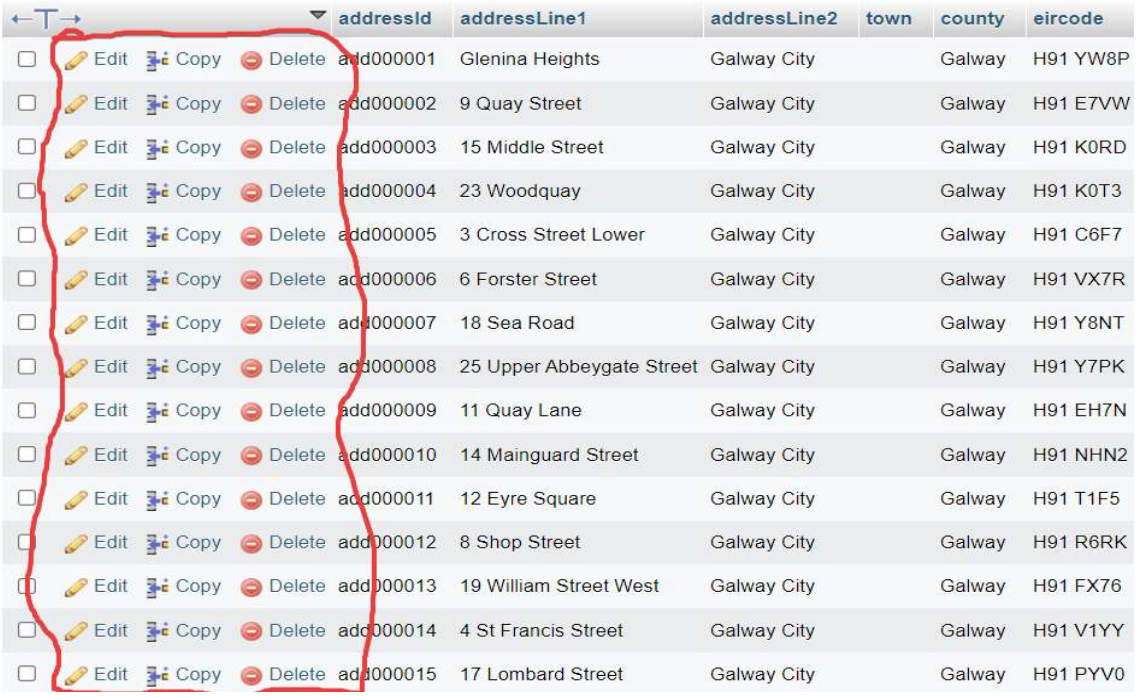
TABLE_CATALOG	TABLE_SCHEMA	TABLE_NAME	TABLE_TYPE	ENGINE	VERSION	ROW_FORMAT	TABLE_ROWS	AVG_ROW_LENGTH
def	information_schema	ALL_PLUGINS	SYSTEM VIEW	Aria	11	Page	NULL	0
def	information_schema	APPLICABLE_ROLES	SYSTEM VIEW	MEMORY	11	Fixed	NULL	979
def	information_schema	CHARACTER_SETS	SYSTEM VIEW	MEMORY	11	Fixed	NULL	384
def	information_schema	CHECK_CONSTRAINTS	SYSTEM VIEW	Aria	11	Page	NULL	0
def	information_schema	COLLATIONS	SYSTEM VIEW	MEMORY	11	Fixed	NULL	231
def	information_schema	COLLATION_CHARACTER_SET_APPLICABILITY	SYSTEM VIEW	MEMORY	11	Fixed	NULL	195
def	information_schema	COLUMNS	SYSTEM VIEW	Aria	11	Page	NULL	0
def	information_schema	COLUMN_PRIVILEGES	SYSTEM VIEW	MEMORY	11	Fixed	NULL	2893
def	information_schema	ENABLED_ROLES	SYSTEM VIEW	MEMORY	11	Fixed	NULL	387
def	information_schema	ENGINES	SYSTEM VIEW	MEMORY	11	Fixed	NULL	731
def	information_schema	EVENTS	SYSTEM VIEW	Aria	11	Page	NULL	0
def	information_schema	FILES	SYSTEM VIEW	MEMORY	11	Fixed	NULL	4022
def	information_schema	GLOBAL_STATUS	SYSTEM VIEW	MEMORY	11	Fixed	NULL	6340
def	information_schema	GLOBAL_VARIABLES	SYSTEM VIEW	MEMORY	11	Fixed	NULL	6340
def	information_schema	KEYWORDS	SYSTEM VIEW	MEMORY	11	Fixed	NULL	194
def	information_schema	KEY_CACHES	SYSTEM VIEW	MEMORY	11	Fixed	NULL	659
def	information_schema	KEY_COLUMN_USAGE	SYSTEM VIEW	MEMORY	11	Fixed	NULL	4637
def	information_schema	OPTIMIZER_TRACE	SYSTEM VIEW	Aria	11	Page	NULL	0
def	information_schema	PARAMETERS	SYSTEM VIEW	Aria	11	Page	NULL	0
def	information_schema	PARTITIONS	SYSTEM VIEW	Aria	11	Page	NULL	0
def	information_schema	PLUGINS	SYSTEM VIEW	Aria	11	Page	NULL	0
def	information_schema	PROCESSLIST	SYSTEM VIEW	Aria	11	Page	NULL	0
def	information_schema	PROFILING	SYSTEM VIEW	MEMORY	11	Fixed	NULL	308
def	information_schema	REFERENTIAL_CONSTRAINTS	SYSTEM VIEW	MEMORY	11	Fixed	NULL	4814
Console	information_schema	ROUTINES	SYSTEM VIEW	Aria	11	Page	NULL	0

Rule 5: Comprehensive Data Sublanguage Rule

"A relational system may support several languages and various modes of terminal use (for example, the fill-in-the-blanks mode). However, there must be at least one language whose statements are expressible, per some well-defined syntax. The database created uses SQL as the data sublanguage, which supports SQL features and allows authorized users to define, manipulate, and query the data as well as define and enforce integrity constraints and authorize access to the database.

Rule 6: View Updating Rule

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



	addressId	addressLine1	addressLine2	town	county	eircode
Edit Copy Delete	add000001	Glenina Heights	Galway City		Galway	H91 YW8P
Edit Copy Delete	add000002	9 Quay Street	Galway City		Galway	H91 E7VW
Edit Copy Delete	add000003	15 Middle Street	Galway City		Galway	H91 K0RD
Edit Copy Delete	add000004	23 Woodquay	Galway City		Galway	H91 K0T3
Edit Copy Delete	add000005	3 Cross Street Lower	Galway City		Galway	H91 C6F7
Edit Copy Delete	add000006	6 Forster Street	Galway City		Galway	H91 VX7R
Edit Copy Delete	add000007	18 Sea Road	Galway City		Galway	H91 Y8NT
Edit Copy Delete	add000008	25 Upper Abbeygate Street	Galway City		Galway	H91 Y7PK
Edit Copy Delete	add000009	11 Quay Lane	Galway City		Galway	H91 EH7N
Edit Copy Delete	add000010	14 Mainguard Street	Galway City		Galway	H91 NHN2
Edit Copy Delete	add000011	12 Eyre Square	Galway City		Galway	H91 T1F5
Edit Copy Delete	add000012	8 Shop Street	Galway City		Galway	H91 R6RK
Edit Copy Delete	add000013	19 William Street West	Galway City		Galway	H91 FX76
Edit Copy Delete	add000014	4 St Francis Street	Galway City		Galway	H91 V1YY
Edit Copy Delete	add000015	17 Lombard Street	Galway City		Galway	H91 PYV0

Rule 7: High-Level Insert, Update, and Delete

A database must support high-level insertion, updating, and deletion. This must not be limited to a single row, that is, it must also support union, intersection and minus operations to yield sets of data records. The database created allows for the insertion, update, and deletion of data at both the base and derived relation levels using SQL commands such as INSERT, UPDATE, and DELETE. For example, a new appointment can be inserted into the Appointment table, or an existing medication record can be updated in the Medication table.

Rule 8: Physical Data Independence

Changes to the physical level should have minimal impact on applications written at the logical level. The database created provides physical data independence, meaning that changes to the physical level, such as adding new columns or changing data types, should have minimal impact on applications written at the logical level. For example, if a new column is added to the Animal table, applications that query data from the Animal table will not need to be modified.

Rule 9: Logical Data Independence

Changes to the logical level (tables, columns, rows, and so on) should have minimal impact on applications written at the conceptual level. The database created provides logical data independence, meaning that changes to the logical level, such as adding new tables or modifying relationships between tables, should have minimal impact on applications written at the conceptual level.

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. The database created defines integrity constraints, such as primary key constraints and foreign key constraints, in the

relational data sublanguage using SQL commands, and they are stored in the catalog. This ensures that the constraints can be enforced by the system rather than by application programs.

Rule 11: Distribution Independence

The end-user must not be able to see that the data is distributed over various locations. Users should always get the impression that the data is located at one site only. This rule has been regarded as the foundation of distributed database systems.

Rule 12: Non-Subversion Rule

If a system has an interface that provides access to low-level records, then the interface must not be able to subvert the system and bypass security and integrity constraints.

References

- "Codd's 12 Rules." Www.tutorialspoint.com, www.tutorialspoint.com/dbms/dbms_codds_rules.htm#. Accessed 16 May 2023.