Nate Flasher

Codd's Twelve Rules

1) PostgreSQL does indeed treat NULL values properly. In PostgreSQL you are able to use a NULL value to specify both unknown values and information that is not available. There are many places we can find NULL values being supported. Some of the few include the use of NULL values in comparisons, constraints, queries, and insert statements. 3 valued logic is also supported in postgres where values can be true, false, and also NULL. Those 3 logic values, including NULL can be compared to one another using logic operands.

   Resources used:
https://www.instaclustr.com/the-postgresql-boolean-three-valued-logic-data-type/
PostgreSQL documentation: https://www.postgresql.org/docs/8.3/functions-comparison.html
https://www.postgresql.org/docs/8.0/ddl-constraints.html
https://www.postgresql.org/docs/8.3/queries-order.html
https://www.postgresql.org/docs/7.3/sql-insert.html

2) *Select* datname *from* pg_database
3)
   a) Postgres supports data definition (https://www.postgresql.org/docs/current/ddl.html) because the user is able to create data by making tables and defining data in relations and putting constraints on such data.
   b) Postgres supports view definition (https://www.postgresql.org/docs/9.2/sql-createview.html) because you are able to create views, including views with parameters.
   c) Data manipulation is supported by postgres (https://www.postgresql.org/docs/9.0/dml.html) because you are able to insert, update, and delete data.
   d) Postgres supports many integrity constraints (https://www.postgresql.org/docs/9.4/ddl-constraints.html) including check, not null, unique, and key constraints.
   e) Postgres supports authorization (https://www.postgresql.org/docs/9.0/auth-methods.html) by having password checks, trust authentications, and more
   f) Transaction management/boundaries are supported by postgres (https://www.postgresql.org/docs/11/spi-transaction.html) as it allows for starting a transaction, committing, and rolling back a transaction
4) Constraints that you are able to implement when creating a table in postgres include the definition of primary and foreign keys, check, conclusion, not null, and unique constraints. (https://www.postgresql.org/docs/9.4/ddl-constraints.html)
5) If the function written in the procedural languages were able to bypass the constraints made for relations/tables/entities at a high level in the database this would compromise

rule 12. However, looking at the postgres documentation (https://www.postgresql.org/docs/current/xplang.html) it seems that there is parsing, syntax analysis, etc that is performed by a handler before being received by the database server. I presume that handler performs a check on the procedural code making sure it does not bypass any high level constraints.