Constraints

Matthew Oresky
Tianhe Wu
Xin Zhang
Project Status

75% Goal:

➔ NOT NULL ✔
➔ UNIQUE for single column ✔
➔ PRIMARY KEY ✔
➔ DEFAULT ✔
➔ Test cases ✔

100% Goal:

➔ CHECK: evaluating simple expressions ✔
➔ End-to-End parser support (mostly) ✔
➔ FOREIGN KEY REFERENCES: implementing foreign key constraint
  ➸ INSERT visibility check ✔
  ➸ CASCADE on DELETE / UPDATE ❓

125% Goal:

➔ UNIQUE multi column ✔ (one step away)
➔ CHECK: supporting all constraint expressions ❓
Explanation

- PRIMARY KEY, UNIQUE → index operations
- CHECK → AbstractExpression parsing
- DEFAULT → default value stored in column definition
- NOT NULL → check at data_table level
- FOREIGN KEY → index operations, visibility check
- Error handling by throwing ConstraintException, message caught at tcop
Foreign Key Explanation

Caveat: Currently we only support creating foreign keys on the Sink’s PK

Foreign Key information: Source table name, Sink table name, Source column(s)

During the execution of a CreatePlan with an FK constraint:
- A Foreign Key constraint object is added to the source => (Sink table name, Source column(s))
- The Source table name is registered with the sink for ON DELETE / ON UPDATE actions
- A non-unique index is made on the source tuples if not NOACTION
Foreign Key cont.

INSERT / UPDATE:
index and visibility check

MATCH FULL?
MATCH PARTIAL?
MATCH SIMPLE?
Foreign Key cont.

NOACTION? CASCADE? SETNULL? SETDEFAULT?

pk1  fk1

pk2  fk2

pk3  fk3

DELETE / UPDATE

visibility check
Demo

```sql
postgres=# CREATE TABLE test (id INT PRIMARY KEY, num INT UNIQUE);
CREATE TABLE test (id INT PRIMARY KEY, num INT UNIQUE) 
postgres=# CREATE TABLE test_table (id INT PRIMARY KEY, num INT UNIQUE, num2 INT NOT NULL);
CREATE TABLE test_table (id INT PRIMARY KEY, num INT UNIQUE, num2 INT NOT NULL);
postgres=# select * from test_table;
  id | num | num2
----------------------
(0 rows)

postgres=# INSERT INTO test_table VALUES (1, 2, NULL);
Not NULL constraint violated : (1, 2, <NULL>)
postgres=# INSERT INTO test_table VALUES (1, 2, 3);
postgres=# INSERT INTO test_table VALUES (1, 2, 3) 1
postgres=# select * from test_table;
  id | num | num2
----------------------
       1 | 2   | 3
(1 row)

postgres=# INSERT INTO test_table VALUES (1, 5, 10);
UNIQUE constraint violated : (1, 5, 10)
postgres=# INSERT INTO test_table VALUES (2, 2, 10);
UNIQUE constraint violated : (2, 2, 10)
postgres=# INSERT INTO test_table VALUES (2, 5, 10);
postgres=# INSERT INTO test_table VALUES (2, 5, 10) 1
postgres=# select * from test_table;
  id | num | num2
----------------------
       1 | 2   | 3
       2 | 5   | 10
(2 rows)
```

Currently supported constraints:
- PRIMARY KEY
- UNIQUE
- NOT NULL
- DEFAULT
- CHECK
Files & Tests

➤ ~/src/catalog: column, constraint and schema definition, table creation
➤ ~/src/storage: data_table.cpp
➤ ~/src/planner: XXX_plan.cpp
➤ ~/src/executor: XXX_executor.cpp
➤ ~/src/parser: create_statement.h
➤ ~/src/tcop: tcop.cpp
➤ ~/test/catalog: constraints_test

➤ NOTNULL_TEST
➤ MULTI_NOTNULL_TEST
➤ CHECK_TEST
➤ DEFAULT_TEST
➤ FOREIGN_KEY_TEST
➤ UNIQUE_TEST
➤ MULTI_UNIQUE_TEST
Performance Test

Constraint Performance Test (No Exception)

- No Constraint
- NOT NULL
- DEFAULT
- CHECK
- NOT NULL + DEFAULT
- NOT NULL + DEFAULT + CHECK

Runtime / sec

50M Tuples (97 MB) 100M Tuples (195 MB)
Issues & Future Improvement

- Foreign key constraint not fully implemented
  - CASCADE
  - Validation of current visibility checking
- Multi-threaded tests needed
- Fine-grained performance test needed
  - Need to know exactly how much we are paying for
- Constraint data structure refactorization
  - In create_statement and catalog
- stats_test problem