Sequences in PostgreSQL

When doing database applications, we sometimes want to generate identifiers automatically, for some rows in the database. The common solution in SQL is to use sequences.

Most DBMSs implement sequences in one form or another, but the syntax may vary from one DBMS to another; in this document we'll focus on sequences in PostgreSQL.

Creating a Sequence

Surprisingly, we use a CREATE SEQUENCE statement to create a sequence, with the name of the sequence (by convention, we end the name in _seq) as in:

```
CREATE SEQUENCE Person_seq;
```

Getting the next value

We can get the value of a sequence by using the function nextval, and passing it the name of the sequence (in single quotes, since the name is a string constant). This function atomically increments the sequence, and returns the old value; if we call it again, we get an incremented value. Notice that even if two users increment the same sequence at 'the same time' the DBMS ensures that the increments are serialized.

We could get the next value in the sequence by using:

```
SELECT NextVal('Person_seq');
```

Default values and SERIAL data tyoe

We can also use NextVal as an expression for the DEFAULT value of a field; if we're doing that, we may prefer to define the field's datatype as SERIAL; PostgreSQL will automatically define a sequence for SERIAL fields, and will automatically insert its next value (incrementing the sequence) when an INSERT statement is issued for that table without a value for that field.

For example, if we define:

```
CREATE TABLE Category (
        Id SERIAL PRIMARY KEY,
        name VARCHAR(20)
);
```

Postgresql will automatically create a sequence (category_seq) for that field. If we insert a row on that table without specifying a value for id, it will automatically use the next value of the sequence; so executing

```
INSERT INTO Category(name) VALUES('One');
INSERT INTO Category(name) VALUES('Two');
```

Will insert two categories, with ids 1 and 2. Notice we could still do something like:

```
INSERT INTO Category(id,name) VALUES(20,'Twenty');
```

but this could cause clashes with future rows that use the sequence.