Autoincrement (identity) is not the only option for primary keys

Jiří Činčura

www.tabsoverspaces.com

Primary key

- *primary key is a specific choice of a minimal set of attributes that uniquely specify a tuple in a relation*
- Unique values
- Not null
- You can define only one for a table

Primary key - practice

- One column
- Easy to use
- Synthetic (can’t be changed, ever)

Autoincrement/Identity

- Numeric value
- Ever increasing

Autoincrement/Identity

Pros:

- Small value
- Simple value (indexing, saving)
- Universally (standard) usable
- “sortable”

Cons:

- Value known only after saving
- Not unique across DB
- Reveals “sequential” information
- Does not work in “distributed” systems
- Can be guessed

UUID/GUID

- 128-bit value
- Can be freely generated

UUID/GUID

Pros:

- Can be generated both on client- and server-side
- Stateless
- Unique across whole DB
- Can be used in distributed systems
- “Cannot” be guessed

Cons:

- Does not have universal representation
- Complex format
- Size
- Difficult saving/indexing
- Version 1 has timestamp value
- Some versions have MAC address
- Unreadable/unsortable
- Can be guessed (security)

Hi/Lo algorithm

- Central entity gives range (lo-hi)
- Client uses range freely
- Unused values are (usually) not returned
- In DBs mostly as a SEQUENCE

Hi/Lo algorithm

Pros:

- Can be generated client- and server-side
- Can be generated across whole DB
- Can be used in distributed systems
- Small value
- Simple value (indexing, saving)
- Universally (standard) usabel
- It is not directly sequential

Cons:

- Step needs to be selected
- Difficult to implement/you need to tools
- Imprecise sorting
- Creates “big gaps”

Conclusion

- There’s no best datatype/value/format
- Theory vs. practice
- Application requirements (and DB)
  - Practical usage

Q & A