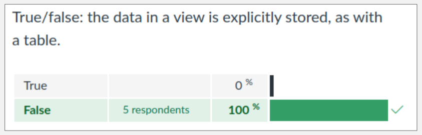
Views

- a *view* is a relation defined by a query over other relations
  - views are virtual – the relation isn't stored, but can be queried as if it is
  - DBMS is responsible for keeping the view up-to-date with changes in the underlying tables


True/false: the data in a view is explicitly stored, as with a table.

True		0 %	
False	5 respondents	100 %	✓

- why use views?
  - convenience, when something complex is referenced often
  - security – can grant permissions for a view without granting access to the underlying tables

Views

- pros
  - data abstraction and simplification
    - can simplify complex queries by abstracting underlying tables into a single virtual table
  - reusability
    - common query structures can be reused across multiple queries
  - provides for logical data independence and a consistent representation of data
    - views help decouple the underlying tables from how the user accesses data
  - supports finer-grained access control
    - can grant access to only specific rows or columns of a table
- cons
  - update limitations
    - UPDATE, INSERT, DELETE limited to certain kinds of views
  - complexity in debugging and management
    - views obscure the underlying dependencies, making it harder to debug and analyze performance
  - performance limits and overhead
    - retrieving data through views can be slower than querying tables directly
    - views often do not have indexes like tables do
  - maintenance overhead
    - changes to underlying tables can break views

Defining a View

```
CREATE VIEW <name> AS
<defn>
```

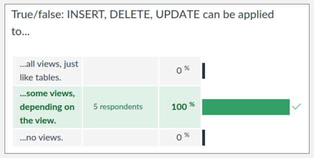
- `<defn>` is a SELECT query that defines the contents of the view
- view name can be used in a SELECT query like any table

```
DROP VIEW <name>
```

- only deletes the definition of the view – underlying tables are not affected

Views – Data Modification

- the rows in a view are automatically updated when the underlying tables are modified


True/false: INSERT, DELETE, UPDATE can be applied to...

...all views, just like tables.		0 %	
...some views, depending on the view.	5 respondents	100 %	✓
...no views.		0 %	

- INSERT, DELETE, UPDATE can be applied to a view in limited circumstances
  - requires an *updateable view* –
    - defined by SELECT (not SELECT DISTINCT)
    - FROM involves only a single base table R (and a single occurrence of R)
    - WHERE does not involve R in a subquery
    - includes all of the NOT NULL attributes of R that don't have default values
    - does not involve aggregation
  - INSERT is passed through to the underlying table with NULL or default values for attributes not present in the view
    - view columns must be simple column references, not expressions
    - if the view query has a WHERE which includes an attribute not in the SELECT, the inserted row may not be visible in the view
  - DELETE, UPDATE are passed through to the underlying table with the addition of the view query's WHERE condition
    - only rows visible in the view are affected

Questions

Do we create views as part of database implementation, or does the application create and drop them as needed?

- views are defined as part of database implementation
  - view definition always exists
- DBMS manages how the view is realized behind the scenes

Questions

How does storage actually work for a virtual table?

- DBMS manages how the view is realized behind the scenes
  - may merge view definition with statement being executed
  - may populate a temporary table
  - some systems support materialized views