A little more about constraints

- A pedagogically great article on constraints (even if it is a trade article)
- Pollack, has been added to box, readings

Purpose [Pollack 2016]

- Prevent bad data from being entered into tables of interest.
- Enforce business logic at the database-level.
- Documentation of important database rules.
- Enforce relational integrity between any number of tables.
- Improve database performance.
- Enforce uniqueness.

Views: A Database Mechanism

1. For expressing a subroutine like mechanism
2. For capturing application semantics
3. For defining alternate data models of the actual DB

Views: A Database Mechanism

1. For expressing a subroutine like mechanism
2. For capturing application semantics
3. For defining alternate data models of the actual DB

Applications (later):

- Data Integration
- Expression and Compilation of Inheritance in UML

Today's Objectives:

- Gross concept
- Syntax and Semantics
- Logical (unmaterialized) vs. Materialized Views

Recall:

- Relational operators take relation(s) as arguments, return a relation as their value.
- SQL query returns a table.

To Define a View:

CREATE VIEW *view-name AS*

select, from, where

"view-name" can then be used nearly anywhere a table name can be used.

Example 1:

Suppose we have a table:

Student(first, last, address, SS#)

CREATE VIEW *view-name AS a SFW query*

We need to be diligent about privacy of SS#s:

```
CREATE VIEW publicStudent AS
SELECT first, last, address
FROM Student
```

```
SELECT *
FROM publicStudent PS
WHERE PS.last = 'Smith'
```

What does this return?
What does it not return?

Semantics: Rewrite as Nested Query

```
SELECT *
FROM publicStudent PS
WHERE PS.last = 'Smith'

CREATE VIEW publicStudent AS
SELECT first, last, address
FROM Student
```

```
SELECT *
FROM
  (SELECT first, last, address
      FROM Student)
      AS PS
WHERE PS.last = 'Smith'
```

2 Things to Observe

1. publicStudent is, effectively, a different model of the data.

   //speaks to being able to restructure the data model

2. As a new model, it hides information.

   // an implementation of privacy and security

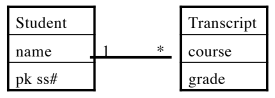
DBA Moment

- Never exploit Social Security number as a key.

DBA Moment:

- Never exploit Social Security number as a key.

Subtlty:



what get's used as the fk?

[Legacy] Three Schema Model - A word about security

A database guys perspective - everything's a schema

- ANSI/SPARC introduced the three schema model in 1975

  // Legacy – in this case, even before my time, ***but it makes the point***

- It provides a framework describing the role and purpose of data modeling

Three Schema Model

Design with Security in Mind

- External schema or user view
  - Representation of how users view the database
- Conceptual schema
  - A logical view of the database containing a description of all the data and relationships
- Internal schema
  - A representation of a conceptual schema as physically stored on a particular product

Example 1, above:

- transform a data model for security

Example 2, next:

- use as a subroutine

Example 2:

```
CREATE VIEW CheapBeer
    SELECT DISTINCT name, b_id
    FROM Beers, Sells
    WHERE Beer.b_Id = Sells.b_Id AND
          Sells.price < (SELECT AVG(price)
                         FROM  Sells);
```

Query: Really Cheap Beer

```
SELECT name
FROM CheepBeer, Sells
WHERE CheepBeer.b_Id = Sells.b_Id AND
      Sells.price < (SELECT AVG(price)
                     FROM  Sells);
```

** CheepBeer are those that are less than the average. The join condition will limit the next calculation to those beers that are less than average cost. i.e. the average of those that are less than average.

Where can I not use a view?

- Consider:

  INSERT INTO publicStudent(‘Jeff’ ,’ Zhu’ ,’ Jester’)

```
CREATE VIEW publicStudent
SELECT first, last, address
FROM Student
```

- Base Relation:

  Student(first, last, address, SS#)

>SS# is missing… can’t do it,
or
- insert NULL ?
- *it depends… vendor*

Implementation Logical vs. Materialized

- Materialize: The view’s query result is computed and saved to disk.

  ** In SQL, reserved keyword, Materialize, inserted as part of view definition.

- What happens when a row is added to base relation

What about?

- Consider:

  INSERT INTO Student(‘Jeff’ , ‘Zhu’ ,’ Jester ‘, 123-45-6789)
  // inserting into a *base relation*

- What happens to?:

```
CREATE VIEW publicStudent
SELECT first, last, address
    FROM Student
```

- It depends on the implementation

~~*Next slide, then*~~
- ~~*Relational Algebra, on board,*~~
- ~~*Where have we seen this before?*~~

Materialized View (Oracle)

CREATE MATERIALIZED VIEW
- Query is executed and result stored

REFRESH MATERIALIZED VIEW
- replaces the contents of a materialized view (re-executes the query)

Incremental Maintenance of a View

- Key axiom: **|X| is join

$$(R \cup S)\ |X|\ T = (R\ |X|\ T) \cup (S\ |X|\ T)$$

- So, if View, V = (S |X| T)
- Insert Into S, rows R, to create S'

  V' = S' |X| T

Incremental Maintenance of a View

- Key axiom: **|X| is join

$$(R \cup S)\ |X|\ T = (R\ |X|\ T) \cup (S\ |X|\ T)$$

Incremental Maintenance of a View

$$(R \cup S)\ |X|\ T = (R\ |X|\ T) \cup (S\ |X|\ T)$$

So, if View, V = (S |X| T)

- Insert Into S, rows R, i.e. S' = $(R \cup S)$
- New view contents, V', V' = S' |X| T

$$\begin{aligned} V' &= (R \cup S)\ |X|\ T \\ &= (R\ |X|\ T) \cup (S\ |X|\ T) \\ &= (R\ |X|\ T) \cup V \end{aligned}$$

Incremental Maintenance of Materialized Views

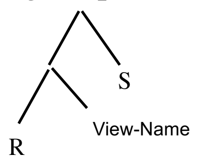
- Incremental axioms exist for all relational operators.
- If interested see qian91 paper in recommended readings.

Logical Implementation

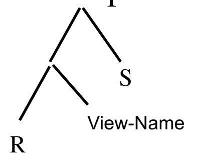
Logical: Query optimizer operates on the plan

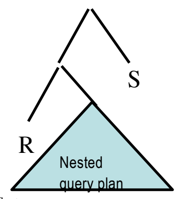
Query plan on board…

- Details will be posted
- Basics, initial logical plan



- Basics, initial logical plan
- Replace View-Name, with logical plan of the view's defining query.
- Let the optimizer do it's job…





Where can I not use a view?

- Consider:

  INSERT INTO publicStudent(‘Jeff’ ,’ Zhu’ ,’ Jester’)

  CREATE VIEW publicStudent
  SELECT first, last, address
  FROM Student

- Base Relation:

  Student(first, last, address, SS#)

>SS# is missing
- insert into Student table the row + NULL ?
- *it depends… vendor*

What about deleting from a view?

- Could work for the examples….
- What about in general?
- What to do? Put the developer in control

Options:
- – programmer can pick flags and their settings
- – allow triggers on views

In Oracle

```
CREATE [OR REPLACE] VIEW [{FORCE | NOFORCE}] VIEW view_name
    [(alias_name[, alias_name...])] AS subquery
    [WITH {CHECK OPTION | READ ONLY} CONSTRAINT constraint_name];
```

http://www.java2s.com/Tutorial/Oracle/0160__View/CreatingandUsingaView.htm

8.WITH CHECK OPTION specifies that only the rows that would be retrieved by the subquery can be inserted, updated, or deleted.
9.By default, rows are not checked that they are retrievable by the subquery before they are inserted, updated, or deleted.
10.constraint_name specifies the name of the WITH CHECK OPTION or READ ONLY constraint.
11.WITH READ ONLY specifies that rows may only read from the base tables.