Defining a Database Schema

CREATE TABLE name (list of elements).

- Principal elements are attributes and their types, but key declarations and constraints also appear.

- Similar CREATE X commands for other schema elements X: views, indexes, assertions, triggers, domains.

  ✦ Assertions and domains not in Oracle 7.3.2.

- “DROP X name” deletes the created element of kind X with that name.

Example

    CREATE TABLE Sells (      
      bar CHAR(20),        
      beer VARCHAR(20),    
      price real          
    );

    DROP TABLE Sells;

Types

1. int or integer.
2. real or float.
3. \texttt{CHAR}(n) = \textit{fixed length character string}, padded with “pad characters.”
4. \texttt{VARCHAR}(n) = \textit{variable-length strings up to} \textit{n characters}.
5. \texttt{BIT}(n) = \textit{bit string of length} \textit{n}.
   ✦ Not in Oracle 7.3.2.
6. Dates. SQL2 form is \texttt{DATE ’yyyy-mm-dd’}
   ✦ Oracle uses a different format — to be explained.
7. Times. Form is \texttt{TIME ’hh:mm:ss[.ss...]’} in SQL2.
Oracle Default Dates (Used at Stanford)

Format ‘dd-mon-yy’

Example

```
CREATE TABLE Days (  
    d DATE
);

INSERT INTO Days  
VALUES('06-nov-97');
```

- Oracle function `to_date` converts a specified format into default format.
```
INSERT INTO Days  
VALUES(to_date('2000-01-01',  
        'yyyy-mm-dd'));
```

- Stored in our system as ‘01-jan-00’.

✦ Now do you believe there is a “year 2000” problem?
Declaring Keys

Use PRIMARY KEY or UNIQUE.

- Oracle 7.3.2 treats these as synonyms.
- But only one primary key, many “uniques” allowed.
- SQL2 allows implementations to create an index (data structure to speed access given a key value) only in response to PRIMARY KEY.
  ♦ But Oracle creates indexes for both.
- Two places to declare:
  1. After an attribute’s type, if the attribute is a key by itself.
  2. As a separate element.
  ♦ Essential if key is > 1 attribute.
Example

```
CREATE TABLE Sells (  
    bar CHAR(20),  
    beer VARCHAR(20),  
    price real,  
    PRIMARY KEY(bar,beer)  
);  
```

- On the Stanford Oracle system for this class, there is a separate data area on a separate disk for indexes.
  - Speeds access — two heads are better than one.
  - Thus, you must follow any implicit index-creating statement like “primary key,” by:

```
USING INDEX TABLESPACE csindx
```
Example

```
CREATE TABLE Beers (  
    name CHAR(20) UNIQUE  
    USING INDEX TABLESPACE csindx,  
    manf CHAR(20)  
);```

Other Properties You Can Give to Attributes

1. **NOT NULL** = every tuple must have a real value for this attribute.

2. **DEFAULT** value = a value to use whenever no other value of this attribute is known.

Example

```sql
CREATE TABLE Drinkers (  
  name CHAR(30) PRIMARY KEY  
    USING INDEX TABLESPACE csindx,  
  addr CHAR(50)  
    DEFAULT '123 Sesame St',  
  phone CHAR(16)  
);```
INSERT INTO Drinkers(name)
VALUES('Sally')

results in the following tuple:

<table>
<thead>
<tr>
<th>name</th>
<th>addr</th>
<th>phone</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sally</td>
<td>123 Sesame St.</td>
<td>NULL</td>
</tr>
</tbody>
</table>

- Primary key is by default not NULL.
- This insert is legal.
  - ✷ OK to list a subset of the attributes and values for only this subset.
- But if we had declared
  
  phone CHAR(16) NOT NULL

then the insertion could not be made.
Changing Columns

Add an attribute of relation $R$ with

$$\text{ALTER TABLE } R \text{ ADD } \langle \text{column declaration} \rangle;$$

Example

$$\text{ALTER TABLE Bars ADD phone CHAR(16) DEFAULT 'unlisted';}$$

- SQL2 allows columns to be dropped, e.g.,

  $$\text{ALTER TABLE Bars DROP license;}$$

- However, this statement is illegal in Oracle 7.3.2.
Views

An expression that describes a table without creating it.

- View definition form is:
  
  `CREATE VIEW <name> AS
  <query> ;`
Example

The view CanDrink is the set of drinker-beer pairs such that the drinker frequents at least one bar that serves the beer.

\[
\text{CREATE VIEW CanDrink AS}
\text{SELECT drinker, beer}
\text{FROM Frequents, Sells}
\text{WHERE Frequents.bar = Sells.bar;}
\]

Querying Views

Treat the view as if it were a materialized relation.

Example

\[
\text{SELECT beer}
\text{FROM CanDrink}
\text{WHERE drinker = 'Sally';}
\]
Semantics of View Use

SQL query $\rightarrow$ rel. algebra $\rightarrow$ SQL

SQL view def. $\rightarrow$ rel. algebra

Example

```
\pi_{drinker,beer} \Join \sigma_{drinker='Sally'} \pi_{beer}
```

Frequents $\rightarrow$ Sells $\rightarrow$ CanDrink $\rightarrow$ Query
Compose

\[ \pi_{\text{beer}} \]
\[ \sigma_{\text{drinker}=\text{'Sally'}} \]
\[ \pi_{\text{drinker,beer}} \]
\[ \bowtie \]

Frequents Sells
Optimize Query

1. Push selections down tree.
2. Eliminate unnecessary projections.

\[ \pi_{beer} \]
\[ \bigotimes \]
\[ \sigma_{drinker='Sally'} \]
\[ \text{Sells} \]
\[ \text{Frequents} \]
Nulls

In place of a value in a tuple’s component.

- Interpretation is not exactly “missing value.”
- There could be many reasons why no value is present, e.g., “value inappropriate.”

Comparing Nulls to Values

- 3rd truth value UNKNOWN.

Example

<table>
<thead>
<tr>
<th>bar</th>
<th>beer</th>
<th>price</th>
</tr>
</thead>
<tbody>
<tr>
<td>Joe’s bar</td>
<td>Bud</td>
<td>NULL</td>
</tr>
</tbody>
</table>

SELECT bar
FROM Sells
WHERE price < 2.00 OR price >= 2.00;

UNKNOWN       UNKNOWN

UNKNOWN
3-Valued Logic

Think of true = 1; false = 0, and unknown = 1/2. Then:

- AND = min.
- OR = max.
- NOT(x) = 1 – x.

Some Key Laws Fail to Hold

Example: Law of the excluded middle, i.e.,

\[ p \text{ OR NOT } p = \text{TRUE} \]

- For 3-valued logic: if \( p = \text{unknown} \), then left side = \( \max(1/2, (1-1/2)) = 1/2 \neq 1 \).
- Like bag algebra, there is no way known to make 3-valued logic conform to all the laws we expect for sets/2-valued logic, respectively.