An Example Database
Special Operators in SQL

- ANSI standard SQL allows the use of special operators in conjunction with the WHERE clause. These special operators (see Day 17, page 26) include:

  BETWEEN – Used to check whether an attribute value is within a range.

  IS NULL – Used to determine if an attribute value is null.

  LIKE – Used to match an attribute value to a string pattern. Many wildcard options are available.

  IN – Used to determine if an attribute value is within a list of values.

  EXISTS – Used to determine if a subquery returns an empty set or not.
The BETWEEN Special Operator

- Suppose that we want to see a listing for all products whose prices are between $50 and $100. The BETWEEN operator can be used for this query expression.

```
SELECT *
FROM PRODUCT
WHERE P_PRICE BETWEEN 50.00 AND 100.00;
```

- If your RDBMS does not support BETWEEN you would need to express this query as:

```
SELECT *
FROM PRODUCT
WHERE P_PRICE > 50.00 AND P_PRICE < 100.00;
```
The IS NULL Special Operator

- Suppose that we want to see a listing for all products that do not currently have a vendor assigned, i.e., V_CODE = null. The null entries could be found with the following query expression.

```
SELECT P_CODE, P_DESCRIPT, V_CODE
FROM PRODUCT
WHERE V_CODE IS NULL;
```

- NOTE: SQL uses a special operator for testing for nulls. You cannot use a condition such as V_CODE = NULL. The reason is that NULL is technically not a “value”, but a special property of an attribute that represents precisely the absence of any value at all.
The LIKE Special Operator

• The LIKE special operator is used in conjunction with wildcards to find patterns within string attributes.

• Standard SQL allows you to use the percent sign (%) and underscore (_) wildcard characters to make matches when the entire string is not known.

   % means any and all following characters are eligible.

   ‘M%’ includes Mark, Marci, M-234x, etc.

   _ means any one character may be substituted for the underscore.

   ‘_07-345-887_’ includes 407-345-8871, 007-345-8875

• Note: Access uses * instead of % and ? instead of _. Oracle searches are case-sensitive, Access searches are not.
The LIKE Special Operator (cont.)

- Suppose that we would like to find all the VENDOR rows for contacts whose last names begin with Smith.

```sql
SELECT V_NAME, V_CONTACT, V_AREACODE, V_PHONE
FROM VENDOR
WHERE V_CONTACT LIKE 'Smith%';
```
The IN Special Operator

• Many queries that would seem to require the use of the logical OR operator can be more easily handled with the help of the special operator IN.

• For example the query:

```sql
SELECT *
FROM PRODUCT
WHERE V_CODE = 21344 OR V_CODE = 24288;
```

can be handled more efficiently with:

```sql
SELECT *
FROM PRODUCT
WHERE V_CODE IN (21344, 24288);
```
The IN Special Operator (cont.)

- The IN operator is especially valuable when it is used in conjunction with subqueries.

- For example, suppose you want to list the V_CODE and V_NAME of only those vendors that actually provide products. In this case, you could use a subquery within the IN operator to automatically generate the value list. The query expression would be:

```
SELECT V_CODE, V_NAME
FROM VENDOR
WHERE V_CODE IN ( SELECT V_CODE
                     FROM PRODUCT);
```

- We’ll look more closely at the IN operator later when we deal more in depth with subqueries.
The EXISTS Special Operator

- The EXISTS operator can be sued whenever there is a requirement to execute a command based on the result of another query. That is, if a subquery returns any rows, then run the main query, otherwise, don’t. We’ll see this operator in more detail when we look at subqueries in more depth.

- For example, suppose we want a listing of vendors, but only if there are products to order. The following query will accomplish our task.

```sql
SELECT *
FROM VENDOR
WHERE EXISTS ( SELECT *
               FROM PRODUCT
               WHERE P_ONHAND <= P_MIN);
```
Advanced DDL Commands in SQL

• In the notes for Day 17 we covered the basics of table creation in SQL, including inserting, deleting, and updating rows in tables.

• Now we’ll look at the SQL commands to alter a table’s structure, by changing attribute characteristics and by adding columns. We’ll also see how to add data into new or modified columns. We’ll also see how to copy tables or parts of tables as well as deleting entire tables.
The ALTER Command

• All changes in the table structure are made using the ALTER command, followed by a keyword that produces the specific change you want to make.

• There are three options for the keyword: ADD, MODIFY, and DROP.
  – **ADD** enables you to add a column to a table.
  – **MODIFY** enables you to change a column’s characteristics.
  – **DROP** allows you to delete a column from a table. Most RDBMSs do not allow you to delete a column from a table, unless that column does not contain any values, because such an action may delete crucial data that are used by other tables.
The ALTER Command (cont.)

- The basic syntax of the ALTER command is:

  ```sql
  ALTER TABLE  tablename
  {ADD | MODIFY}(
    columnname  datatype
    [ {ADD | MODIFY} columnname  datatype ]
  );
  ```

- The ALTER TABLE command can also be used to add table constraints. In that case the syntax would be as follows:

  ```sql
  ALTER TABLE  tablename
  ADD  constraint  [ ADD constraint ];
  ```
The ALTER Command (cont.)

- You can also use the ALTER TABLE command to remove a column or table constraint. The basic syntax of this form of the ALTER command is:

```
ALTER TABLE  tablename
DROP { PRIMARY KEY | COLUMN  columnname | CONSTRAINT  constraintname } ;
```

- Notice that when removing a constraint, you need to specify the name given to the constraint. This is one reason why it is always advisable to name the constraints in the CREATE TABLE or ALTER TABLE statements.
Changing a Column’s Data Type

- You can also use the ALTER TABLE command to change the data type of a column.

- For example, suppose we wanted to change the data type of V_CODE attribute in the PRODUCT table from integer to character. The following SQL command would accomplish this task:

```
ALTER TABLE PRODUCT
MODIFY (V_CODE CHAR(5));
```

- Most RDBMSs do not allow you to change the data type of an attribute unless the column to be changed is empty. For example, if we issued the SQL command above on the database we’ve been using and error message would be generated because the V_CODE column already contains data. The reason for the error is simple. The V_CODE attribute in PRODUCT references the V_CODE attribute in VENDOR. If the data types don’t match, there is a referential integrity violation, thus triggering the error message. If the V_CODE column in PRODUCT was empty and the foreign key reference was not specified during the PRODUCT table’s creation, then the SQL command above would execute properly.
Changing a Column’s Data Characteristics

• If the column to be changed already contains data, you can make any changes in the column’s characteristics if those changes do not alter the data type.

• For example, if we wanted to increase the width of the P_PRICE column from 8 digits to 9 digits, we would need to issue the following command:

  ALTER TABLE PRODUCT
  modifying (P_PRICE DECIMAL(9,2));

• Many RDBMSs place restrictions on the types of changes to column characteristics that can occur. For example, Oracle will allow you to widen a column but will not allow you to narrow a column.
Adding a Column to a Table

- You can alter an existing table by adding one or more columns.

- For example, suppose that we want to add a column to the PRODUCT table called P_SALECODE that will allow us to decide if products that have been in inventory for a certain length of time should be placed on sale. Let’s assume that the P_SALECODE entries will be 1, 2, or 3, and we’re not going to do arithmetic on the column so we’ll make it a character.

```sql
ALTER TABLE PRODUCT
ADD (P_SALECODE CHAR(1));
```
Adding a Column to a Table (cont.)

• When adding a column, be careful not to include the NOT NULL clause for the new column. Doing so will cause an error message because when adding a new column to a table that already has rows, the existing rows will default to a value of null for the new column. Therefore, it is not possible to add the NOT NULL clause for this new column.

• You can add the NOT NULL clause to the table structure after all the data for the new column has been entered and the column no longer contains nulls.
Deleting a Column from a Table

- Occasionally, you may want to modify a table by deleting a column.

- Suppose that we want to delete the V_ORDER attribute from the VENDOR table. To accomplish this task we would use the following SQL command:

```
ALTER TABLE VENDOR
  DROP COLUMN V_ORDER;
```

- As before, some RDBMSs will impose restrictions on the deletion of an attribute. For example, most RDBMSs will not allow you to delete attributes that are involved in foreign key relationships, nor may you delete an attribute of a table that contains only that one attribute.
Advanced Data Updates

• To make data entries in an existing row’s columns, SQL uses the UPDATE command. The UPDATE command only updates data in existing rows.

• For example, to enter the P_SALECODE value of ‘2’ in the fourth row of the PRODUCT_2 table, we need to use the UPDATE command in conjunction with the primary key value for that row. The following command produces this effect with the before and after results shown on the next page.

```
UPDATE PRODUCT_2
SET P_SALECODE = '2'
WHERE P_CODE = '1546-QQ2';
```
Advanced Data Updates (cont.)

P_SALECODE value before update
### Advanced Data Updates (cont.)

**P_SALECODE value after update**

![Microsoft Access database](image)

#### PRODUCT_2: Table

<table>
<thead>
<tr>
<th>P_CODE</th>
<th>P_DESCR</th>
<th>P_INDATE</th>
<th>P_ONHAND</th>
<th>P_MIN</th>
<th>P_PRICE</th>
<th>P_DISCOUNT</th>
<th>V_CODE</th>
<th>P_SALECODE</th>
</tr>
</thead>
<tbody>
<tr>
<td>11GRE031</td>
<td>Power painter, 15 psi, 3-nozzle</td>
<td>03-Nov-03</td>
<td>8</td>
<td>5</td>
<td>$109.99</td>
<td>0.00</td>
<td>25595</td>
<td></td>
</tr>
<tr>
<td>13-Q2P2</td>
<td>7.25-in. pwr. saw blade</td>
<td>13-Dec-03</td>
<td>32</td>
<td>15</td>
<td>$14.99</td>
<td>0.05</td>
<td>21344</td>
<td></td>
</tr>
<tr>
<td>14-Q1L3</td>
<td>9.00-in. pwr. saw blade</td>
<td>13-Nov-03</td>
<td>18</td>
<td>12</td>
<td>$17.49</td>
<td>0.00</td>
<td>21344</td>
<td></td>
</tr>
<tr>
<td>1546-QQ2</td>
<td>Hrd. cloth, 1/4-in., 2x50</td>
<td>15-Jan-04</td>
<td>15</td>
<td>8</td>
<td>$39.95</td>
<td>0.00</td>
<td>23119</td>
<td></td>
</tr>
<tr>
<td>1556-QW1</td>
<td>Hrd. cloth, 1/2-in., 3x50</td>
<td>15-Jan-04</td>
<td>23</td>
<td>5</td>
<td>$43.99</td>
<td>0.00</td>
<td>23119</td>
<td></td>
</tr>
<tr>
<td>2232QTS</td>
<td>B&amp;D jigsaw, 12-in. blade</td>
<td>30-Dec-03</td>
<td>5</td>
<td>0</td>
<td>$109.92</td>
<td>0.05</td>
<td>24288</td>
<td></td>
</tr>
<tr>
<td>2232QME</td>
<td>B&amp;D jigsaw, 8-in. blade</td>
<td>24-Dec-03</td>
<td>6</td>
<td>5</td>
<td>$99.87</td>
<td>0.05</td>
<td>24288</td>
<td></td>
</tr>
<tr>
<td>2238QPD</td>
<td>B&amp;D cordless drill, 1/2-in.</td>
<td>20-Jan-04</td>
<td>12</td>
<td>5</td>
<td>$38.95</td>
<td>0.05</td>
<td>25595</td>
<td></td>
</tr>
<tr>
<td>23109-HB</td>
<td>Claw hammer</td>
<td>20-Jan-04</td>
<td>23</td>
<td>10</td>
<td>$9.95</td>
<td>0.10</td>
<td>21225</td>
<td></td>
</tr>
<tr>
<td>23114-AA</td>
<td>Sledge hammer, 12 lb.</td>
<td>02-Jan-04</td>
<td>8</td>
<td>5</td>
<td>$14.40</td>
<td>0.05</td>
<td>21344</td>
<td></td>
</tr>
<tr>
<td>54773-2T</td>
<td>Rat-tail file, 1/8-in. fine</td>
<td>15-Dec-03</td>
<td>43</td>
<td>20</td>
<td>$4.99</td>
<td>0.00</td>
<td>21344</td>
<td></td>
</tr>
<tr>
<td>89-VARE-Q</td>
<td>Hicut chain saw, 16 in.</td>
<td>07-Feb-04</td>
<td>11</td>
<td>5</td>
<td>$256.99</td>
<td>0.05</td>
<td>24288</td>
<td></td>
</tr>
<tr>
<td>PVC33DR</td>
<td>PVC pipe, 3.5-in., 8-ft</td>
<td>20-Feb-04</td>
<td>186</td>
<td>75</td>
<td>$5.87</td>
<td>0.00</td>
<td>21225</td>
<td></td>
</tr>
<tr>
<td>SM-13277</td>
<td>1/2-in. metal screw, 25</td>
<td>01-Mar-04</td>
<td>172</td>
<td>75</td>
<td>$6.99</td>
<td>0.00</td>
<td>21225</td>
<td></td>
</tr>
</tbody>
</table>

*Product description*
Advanced Data Updates (cont.)

• Subsequent data can be entered in the same fashion, defining each entry location by its primary key (P_CODE) and its column location (P_SALECODE).

• The table on the previous page, also reflects the execution of the following SQL command:

```sql
UPDATE PRODUCT_2
    SET P_SALECODE = '1'
    WHERE P_CODE IN ('2232/QWE', '2232/QTY');
```
Advanced Data Updates (cont.)

• Although the UPDATE sequences we’ve just seen allow you to enter values into specified table cells, the process is very cumbersome. Fortunately, there may be better method.

• If a relationship can be established between entries and the existing columns, that relationship may be used to assign values to their appropriate slots.

• For example, suppose that we want to place sales codes based on the P_INDATE into the table using the following rule:
  • If P_INDATE is before December 25, 2003, then P_SALECODE = 2
  • If P_INDATE between Jan. 16 and Feb 10, 2004, then P_SALECODE = 1
Advanced Data Updates (cont.)

• Given the rule from the previous page and using the PRODUCT table to effect the update, the following two command sequences will make all the appropriate updates to the P_SALECODE in the PRODUCT table. The effect of these two updates on the PRODUCT table is shown on the next page.

```
UPDATE PRODUCT
    SET P_SALECODE = '2'
    WHERE P_INDATE < '25-Dec-2003';

UPDATE PRODUCT
    SET P_SALECODE = '1'
    WHERE P_INDATE >= '16-Jan-2004'
        AND P_INDATE <= '10-Feb-2004';
```
### Advanced Data Updates (cont.)

#### Microsoft Access

**PRODUCT_3 : Table**

<table>
<thead>
<tr>
<th>P_CODE</th>
<th>P_DESCRIPC</th>
<th>P_INDATE</th>
<th>P_ONHAND</th>
<th>P_MIN</th>
<th>P_PRICE</th>
<th>P_DISCOUNT</th>
<th>V_CODE</th>
<th>P_SALECODE</th>
</tr>
</thead>
<tbody>
<tr>
<td>11QER31</td>
<td>Power painter, 15 psi., 3-nozzle</td>
<td>03-Nov-03</td>
<td>8</td>
<td>5</td>
<td>$109.99</td>
<td>0.00</td>
<td>25595</td>
<td>2</td>
</tr>
<tr>
<td>13-Q2P2</td>
<td>7.25-in. pwr. saw blade</td>
<td>13-Dec-03</td>
<td>32</td>
<td>15</td>
<td>$14.99</td>
<td>0.05</td>
<td>21344</td>
<td>2</td>
</tr>
<tr>
<td>14-Q1L3</td>
<td>9.00-in. pwr. saw blade</td>
<td>13-Nov-03</td>
<td>18</td>
<td>12</td>
<td>$17.49</td>
<td>0.00</td>
<td>21344</td>
<td>2</td>
</tr>
<tr>
<td>1546-QQ2</td>
<td>Hrd. cloth, 1/4-in., 2x50</td>
<td>15-Jan-04</td>
<td>15</td>
<td>8</td>
<td>$39.95</td>
<td>0.00</td>
<td>23119</td>
<td>2</td>
</tr>
<tr>
<td>1558-QW1</td>
<td>Hrd. cloth, 1/2-in., 3x50</td>
<td>15-Jan-04</td>
<td>23</td>
<td>5</td>
<td>$43.99</td>
<td>0.00</td>
<td>23119</td>
<td>2</td>
</tr>
<tr>
<td>2232-QTY</td>
<td>B&amp;D jigsaw, 12-in. blade</td>
<td>30-Dec-03</td>
<td>8</td>
<td>5</td>
<td>$109.92</td>
<td>0.05</td>
<td>24288</td>
<td>2</td>
</tr>
<tr>
<td>2232-QWE</td>
<td>B&amp;D jigsaw, 8-in. blade</td>
<td>24-Dec-03</td>
<td>6</td>
<td>5</td>
<td>$99.87</td>
<td>0.05</td>
<td>24288</td>
<td>2</td>
</tr>
<tr>
<td>2238-QPD</td>
<td>B&amp;D cordless drill, 1/2-in.</td>
<td>20-Jan-04</td>
<td>12</td>
<td>5</td>
<td>$38.95</td>
<td>0.05</td>
<td>25595</td>
<td>1</td>
</tr>
<tr>
<td>23109-HB</td>
<td>Claw hammer</td>
<td>20-Jan-04</td>
<td>23</td>
<td>10</td>
<td>$9.95</td>
<td>0.10</td>
<td>21225</td>
<td>1</td>
</tr>
<tr>
<td>23114-AA</td>
<td>Sledge hammer, 12 lb.</td>
<td>02-Jan-04</td>
<td>6</td>
<td>5</td>
<td>$14.40</td>
<td>0.05</td>
<td>21225</td>
<td>1</td>
</tr>
<tr>
<td>54778-2T</td>
<td>Rat-tail file, 1/8-in. fine</td>
<td>15-Dec-03</td>
<td>43</td>
<td>20</td>
<td>$4.99</td>
<td>0.00</td>
<td>21344</td>
<td>2</td>
</tr>
<tr>
<td>89-VRE-Q</td>
<td>Hicat chain saw, 16 in.</td>
<td>07-Feb-04</td>
<td>11</td>
<td>5</td>
<td>$259.99</td>
<td>0.05</td>
<td>24288</td>
<td>1</td>
</tr>
<tr>
<td>PVC23DRT</td>
<td>PVC pipe, 3.5-in., 8-ft</td>
<td>20-Feb-04</td>
<td>188</td>
<td>75</td>
<td>$5.87</td>
<td>0.00</td>
<td>21225</td>
<td>2</td>
</tr>
<tr>
<td>SM-18277</td>
<td>1.25-in. metal screw, 25</td>
<td>01-Mar-04</td>
<td>172</td>
<td>75</td>
<td>$6.99</td>
<td>0.00</td>
<td>21225</td>
<td>2</td>
</tr>
<tr>
<td>SW-23116</td>
<td>2.5-in. wd. screw, 50</td>
<td>24-Feb-04</td>
<td>237</td>
<td>100</td>
<td>$8.45</td>
<td>0.00</td>
<td>21231</td>
<td>1</td>
</tr>
<tr>
<td>VR3FIT3</td>
<td>Steel matting, 4&quot;x8&quot;x1/4&quot;, 5&quot; mesh</td>
<td>17-Jan-04</td>
<td>18</td>
<td>5</td>
<td>$119.95</td>
<td>0.10</td>
<td>25595</td>
<td>1</td>
</tr>
</tbody>
</table>

Product code: Primary key

---

COP 4710: Database Systems (Day 18)  Page 26  Mark Llewellyn
Advanced Data Updates (cont.)

• The arithmetic operators are particularly useful in data updates.

• For example, if the quantity on hand in the PRODUCT table has dropped below the minimum desirable value, you’ll order more of the product. Suppose that we’ve ordered 20 more units of product 2232/QWE. When the 20 units arrive, we’ll need to add them to the inventory in the following manner:

```
UPDATE PRODUCT
    SET P_ONHAND = P_ONHAND + 20
WHERE P_CODE = '2232/QWE/;
```
• Suppose that you want to add 10 percent to the price of all products that have current prices below $50. This can be accomplished with the following command.

```sql
UPDATE PRODUCT
  SET P_PRICE = P_PRICE * 1.10
WHERE P__PRICE < 50.00;
```
Copying Parts of Tables

- Although the database should be properly designed before it is implemented, there are times when it may be necessary to break up a table structure into several component parts (smaller tables).
- SQL allows for the copying of the contents of selected table columns so that the data in those columns need not be reentered manually into the newly created table(s).
- Suppose, for example, that we want to copy the P_CODE, P_DESCRIPT, and P_PRICE columns from the PRODUCT table to a new table named PART.
- First, we’ll need to create the PART table as shown on the next page.

```sql
UPDATE PRODUCT
SET P_PRICE = P_PRICE * 1.10
WHERE P_PRICE < 50.00;
```
Copying Parts of Tables (cont.)

CREATE TABLE PART (  
  PART_CODE CHAR(8) NOT NULL UNIQUE,  
  PART_DESCRIPT CHAR(35),  
  PART_PRICE DECIMAL(8,2),  
  PRIMARY KEY (PART_CODE) );

- Note that the PART column names need not be identical to those in the original table, and that the new table need not have the same number of columns as the original table.
  - In this case, the first column in the PART table is PART_CODE, rather than the original P_CODE found in the PRODUCT table. And the PART table contains only three columns, rather than the seven columns found in the PRODUCT table.
  - However, column characteristics must match: you cannot copy a character-based attribute into a numeric structure, and vice versa.
Copying Parts of Tables (cont.)

- Next, we need to add the rows to the new PART table using the PRODUCT table rows. To do this we’ll need to use the INSERT command.

- Recall, the basic syntax of this command:

```sql
INSERT INTO target_tablename [(target_columnlist)]
SELECT source_columnlist
FROM source-tablename;
```

- The target-columnlist is requires if the source-columnlist doesn’t match all of the attribute names and characteristics of the target table (including the order of the columns!). Otherwise, you do not need to specify the target-columnlist.
  - In our example, we will need to specify the target-columnlist since we are changing the names of one or more of the columns.
Copying Parts of Tables (cont.)

- To effect the desired update, we need to issue the following INSERT command:

```
INSERT INTO PART (PART_CODE, PART_DESCRIPT, PART_PRICE)
SELECT P_CODE, P_DESCRIPT, P_PRICE
FROM PRODUCT;
```

- The contents of the updated PARTS table can be viewed with the following query:

```
SELECT *
FROM PART;
```

- The results are shown on the next page.
Copying Parts of Tables (cont.)

Results of copying parts of a table with renaming of the columns.
Adding Primary and Foreign Key Designations

• Although we were able to create a new table based on an existing table in the previous example, the process is not without its problems. Primarily, the new PART table was created without the inclusion of the integrity rules of the table on which it was based. In particular, there is no primary key designated for the table shown on the previous page.

• To define the primary key for this table we need to use the ALTER command as shown below:

```
ALTER TABLE PRODUCT
ADD PRIMARY KEY (P_CODE);
```
Adding Primary and Foreign Key Designations (cont.)

- Quite aside from the fact that the integrity rules are not automatically transferred to a new table that derives its data from one or more other tables, there are several other scenarios that would leave you without entity and referential integrity enforcement.

- For example, you might have simply forgotten to define the primary and foreign keys when you created the tables.

- The integrity rules can be reestablished via the ALTER command as shown below:

```sql
ALTER TABLE PRODUCT
ADD PRIMARY KEY(P_CODE)
ADD FOREIGN KEY(V_CODE) REFERENCES VENDOR;
```
Deleting a Table From the Database

- A table can be deleted from the database through the DROP command as shown below:

  DROP TABLE PRODUCT

- A table can only be dropped from a database if it is not participating as the “1” side of any relationships. If you attempt to delete such a table, the RDMS will issue an error message to indicate that a foreign key integrity violation has occurred.