Database Design - Section 12

Instructor Guide
Table of Contents

Database Design - Section 12 ..........................................................................................................................1
Lesson 1 - The Changing Nature of Work ......................................................................................................1
What Will I Learn? ........................................................................................................................................2
Why Learn It? .............................................................................................................................................3
Try It / Solve It ...........................................................................................................................................4
Lesson 2 - Basic Mapping: The Transformation Process ..............................................................................10
What Will I Learn? ......................................................................................................................................12
Why Learn It? .............................................................................................................................................13
Tell Me / Show Me......................................................................................................................................14
Try It / Solve It ...........................................................................................................................................21
Lesson 3 - Relationship Mapping ................................................................................................................23
What Will I Learn? ......................................................................................................................................24
Why Learn It? .............................................................................................................................................25
Tell Me / Show Me......................................................................................................................................26
Try It / Solve It ...........................................................................................................................................37
Lesson 4 - Subtype Mapping .......................................................................................................................39
What Will I Learn? ......................................................................................................................................42
Why Learn It? .............................................................................................................................................43
Tell Me / Show Me......................................................................................................................................44
Try It / Solve It ...........................................................................................................................................55
Lesson 5 - Introduction to HTML DB: Music Tables ...................................................................................59
What Will I Learn? ......................................................................................................................................62
Why Learn It? .............................................................................................................................................63
Tell Me / Show Me......................................................................................................................................64
Try It / Solve It ...........................................................................................................................................65
Lesson 1 - The Changing Nature of Work

Lesson Preparation
For the Activity, you can direct students to the following links:
(1) http://www.bls.gov/emp/emptab3.htm
(2) http://www.bls.gov/emp/emptab5.htm
These are links to the U.S. Department of Labor, Bureau of Labor Statistics. Outside the U.S., provide the website link that is relevant to your location.

What to Watch For
None.

Connections
Discuss how the school systems have changed over the years, from online systems (registration, exams, classes), to sports (rising popularity of girls’ soccer and basketball teams), to special-interest clubs (clubs for different ethnic groups), etc.
What does this mean for the careers available in a school?
Possible responses include: hiring database administrators and developers, women coaches for sports teams, rising need for people with multilingual skills, etc.
What Will I Learn?

In this lesson, you will learn to:

- Research the factors that contribute to the changing nature of the job market.
- Identify five jobs or job areas that are on the decline and explain why.
- Identify five jobs or job areas that are growing and explain why.
- Articulate the changing nature of work and its associated educational requirements.
- Evaluate career choices in light of the changing nature of work.
Why Learn It?

Pretend you are planning to go on vacation to a tiny island in the Pacific. You save your money, buy your plane ticket, and arrive at the island. Because you didn’t watch the weather, you find the island devastated by a hurricane and your vacation ruined. This is similar to what could happen if you spend time preparing for a job that may not exist after you finish school.
Tell Me / Show Me

How many of you...

- Use an Atari computer to play games?
- Use an electric typewriter to do your homework?
- Watch television on a black and white screen?
- Play music on a record player?
- Know someone who has an 8-track tape?

Thirty years ago the companies that made these products employed many workers to design and produce them. All are near-extinct technologies today.
Tell Me / Show Me

Tell Me / Show Me

Think of a new technology that has or will replace each of the following:

- Hand-crank windows in cars
- Film-based cameras
- Cars that burn gas
- Non-mechanical stores
- Video tapes
- Floppy disks
- CRT monitors
- Light switch

Tell Me / Show Me

Possible answers are:
- Automatic windows
- Digital cameras
- Hybrid or electric cars
- Online stores and music download sites
- DVDs, VCDs
- CDs
- Plasma screens (in case students ask, CRT stands for cathode ray tube -- this is a very old type of computer screen; you could only display text and it was in one color)
- Automatic light (motion) sensors
Tell Me / Show Me

Why would a worldwide burger chain offer vegetarian entrees in some countries?
Answer: To meet the needs of the local market.

Why do some high-end clothes designers create separate clothing labels that are more affordably priced?
Answer: To compete with the discount and bargain stores that sell knockoffs of designer clothes.

Have students check their coats or backpacks or shoes. Where are they made or assembled?
What does this mean for the job market?
Answer: Very few items are totally manufactured in one country. A car is designed in Germany, some parts are made in the U.S., the steel can come from different countries, and it is assembled in Mexico and sold worldwide. This means that you should pay attention to what your own country contributes to global production. This is where a lot of jobs will be. However, this constantly changes, so you should keep track of current trends.
Tell Me / Show Me

Describe the different ways in which technology has affected employment and industry. Provide specific examples of industries and occupations that have been impacted by technology.

Possible responses include:

- The electric-typewriter manufacturers and the service industry that employed workers to repair them.
  - Workers had to learn to repair newer machines and computers.
- Companies that designed and made television antennas.
  - Needed to learn newer technology, such as satellite dishes or cable TV.
- DNA profiling and testing.
  - Created new jobs in police departments for scientists familiar with DNA procedures.
Tell Me / Show Me

You could also point out that baby boomers born after World War II are now approaching old age, increasing the demand for different products and services such as expanded health care, retirement homes, and medicines. These demands create new technologies, new products, and a new skilled workforce.
Try It / Solve It

Possible responses for “fastest-growing” list:
1. Database administrators: As companies automate their information-processing systems, databases are needed to contain and manage the information.
2. Hazardous-materials-removal workers: As the need for cleaning up and preserving the environment grows, professionals who know how to deal with hazardous materials will be in demand.
3. Fitness trainers and aerobics instructors: As obesity becomes a bigger problem, fitness becomes a more important issue.

Possible responses for “declining” list:
1. Postal-service mail sorters, processors, and processing-machine operators: To be replaced with email and electronic transfer of information.
2. Meter readers for utilities: Gas/water/electrical usage will be electronically recorded.
3. Bank tellers: Electronic online banking services.
Lesson 2 - Basic Mapping: The Transformation Process

Lesson Preparation

None.

What to Watch For

Students may not see a great difference between the conceptual model and the relational design at this point. They may ask why not create table definitions from the start. Explain that simple entities (like the ones they just transformed in the activity) are very similar to relational tables. However, once we get to foreign keys, arcs, and supertypes, there will be differences.

Also, remind them that when they create a conceptual model, they are focused on the business and its rules. When they create a database design, the focus will be on database issues of storage, speed of transactions, security, etc. Although these are important issues, they should not be considered before or above the business requirements. Data modeling pays attention to the business requirements, regardless of implementation. You may have the fastest and most secure database in the world, but if it doesn’t meet your business requirements, it’s not going to be of much use.

Connections

Recall previous lesson(s) where students modeled an animal shelter. Tell students to imagine that the director approved their model and wanted a system built based on it, using a relational
database. Transforming that model into a database design would be the next phase. This lesson teaches them the first steps in that phase.
What Will I Learn?

In this lesson, you will learn to:

- Distinguish entity-relationship models from database models
- Describe the terminology mapping between a conceptual model and a relational database model
- Apply the rule of basic mapping to transform an entity into a table
- Apply the rule of Oracle naming conventions for tables and columns used in relational models
Why Learn It?

When you design a house, you eventually would like to see the house built. Even if you don’t do the actual construction, you will need to understand the terms used by the builders in order to help them take your design and make it a reality.

This “first-cut” design can be used for further discussion between designers, developers, and database administrators.
This is a review of the concepts covered in Section 11, Lesson 5.
Tell Me / Show Me

Differentiate between the EMPLOYEE entity in the ERD (conceptual model) and the diagram of the EMPLOYEES table, which represents the definition of the table in the relational model (physical implementation). The notations in the table diagram will be explained throughout this lesson.
Explain that analysis and design are phases of the system life cycle (to be discussed more later). When designing a system, analysis precedes design. Data modeling is done in the analysis phase. When you are satisfied that you have captured the business requirements in the data model, you move on to the design phase, where the ERD is mapped to a physical implementation.
Tell Me / Show Me

Section 12
Lesson 2 - Basic Mapping: The Transformation Process

Tell Me / Show Me

Table Diagram Notations

The first row of the table diagram contains the table name and the short name.

The Key Type column may contain values of "pk" for the primary key, "uk" for the unique key, and "fk" for the foreign key column. It may also be blank if the column is not a part of any key.

The Optionality column must contain "**" if the column is mandatory and "*" if it is optional. This is similar to the entity diagram.

The third column is for the column name.
### Naming Conventions for Tables and Columns

- The table name is the plural of the entity name.
- Example: `STUDENT` becomes `STUDENTS`.
- Column names are identical to the attribute names, except that special characters and spaces are replaced with underscores. Column names also often use more abbreviations than attribute names.
- Example: `first name` becomes `first_name`, or `fn` if it's a primary key.

#### STUDENTS

<table>
<thead>
<tr>
<th>Key type</th>
<th>Optionality</th>
<th>Column Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>pk</td>
<td>*</td>
<td>id</td>
</tr>
<tr>
<td></td>
<td>*</td>
<td>first_name</td>
</tr>
<tr>
<td></td>
<td>*</td>
<td>last_name</td>
</tr>
<tr>
<td></td>
<td>*</td>
<td>street_address</td>
</tr>
<tr>
<td></td>
<td>*</td>
<td>city</td>
</tr>
<tr>
<td></td>
<td>*</td>
<td>state</td>
</tr>
<tr>
<td></td>
<td>*</td>
<td>p_code</td>
</tr>
<tr>
<td></td>
<td>*</td>
<td>dob</td>
</tr>
</tbody>
</table>
Tell Me / Show Me

These rules do not guarantee uniqueness, but experience has proved that duplicated names are relatively rare. In the case of identical short names, just add a number to the one that is used less. Example: CTR and CTR1.
Tell Me / Show Me

Tell Me / Show Me

Naming Restrictions with Oracle

- Table and column names:
  - must start with a letter
  - can contain up to 30 alphanumeric characters
  - cannot contain space or special characters such as "+", "/", "#", and "'" are permitted

- Table names must be unique.

- Column names must be unique within a table.

- Some words have a special meaning in the Oracle database and in the programming language. These are called "reserved" words. It is best to avoid using these as names for your tables and columns. Some common examples of Oracle reserved words are:
  - NUMBER
  - SEQUENCE
  - VALUES
  - LEVEL
  - TYPE

A complete list can be found on Technet.

Tell Me / Show Me

Show the students how to use Technet where the lists of reserved words in Oracle 10g can be found. This site requires you to sign up, but it is free. It's a valuable source of technical information on all Oracle products.

Explain that all database systems make recommendations on naming objects (such as tables). If they do not use an Oracle database, they should still decide on a naming convention and make sure it is compatible with the database system that they have chosen.
Try It / Solve It

Activity: Mapping a Simple Entity

Transform the following entities in Global Fast Foods into table definitions:

SHIFT
REGULAR MENU
PROMOTIONAL MENU
FREQUENT DINER CARD

These are called simple entities because they are on the one side of a 1:M relationship.

Use as many rows as necessary in the following table diagram.

<table>
<thead>
<tr>
<th>TABLE NAME (short name)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Key Type (pk, uk, fk)</td>
</tr>
</tbody>
</table>

Try It / Solve It

Explain that we are not creating tables; we are creating definitions of tables. The physical implementation of the table will be done in a later lesson using HTML DB.
<table>
<thead>
<tr>
<th>Key Type</th>
<th>Optionality</th>
<th>Column Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>pk</td>
<td>*</td>
<td>code</td>
</tr>
<tr>
<td></td>
<td>*</td>
<td>description</td>
</tr>
</tbody>
</table>

**REGULAR_MENUS (RMU)**

<table>
<thead>
<tr>
<th>Key Type</th>
<th>Optionality</th>
<th>Column Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>pk</td>
<td>*</td>
<td>code</td>
</tr>
<tr>
<td></td>
<td>*</td>
<td>rm_type</td>
</tr>
<tr>
<td></td>
<td>*</td>
<td>hours_served</td>
</tr>
</tbody>
</table>

**PROMOTIONAL_MENUS (PMU)**

<table>
<thead>
<tr>
<th>Key Type</th>
<th>Optionality</th>
<th>Column Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>pk</td>
<td>*</td>
<td>code</td>
</tr>
<tr>
<td></td>
<td>*</td>
<td>name</td>
</tr>
<tr>
<td></td>
<td>*</td>
<td>start_date</td>
</tr>
<tr>
<td></td>
<td>*</td>
<td>end_date</td>
</tr>
<tr>
<td></td>
<td>*</td>
<td>giveaway</td>
</tr>
</tbody>
</table>

**FREQ_DIN_CRDS (FDD)**

<table>
<thead>
<tr>
<th>Key Type</th>
<th>Optionality</th>
<th>Column Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>pk</td>
<td>*</td>
<td>id</td>
</tr>
<tr>
<td></td>
<td>*</td>
<td>first_name</td>
</tr>
<tr>
<td></td>
<td>*</td>
<td>last_name</td>
</tr>
<tr>
<td></td>
<td>*</td>
<td>address</td>
</tr>
<tr>
<td></td>
<td>*</td>
<td>city</td>
</tr>
<tr>
<td></td>
<td>*</td>
<td>state</td>
</tr>
<tr>
<td></td>
<td>*</td>
<td>zip</td>
</tr>
<tr>
<td></td>
<td>*</td>
<td>phone_nbr</td>
</tr>
</tbody>
</table>
Lesson 3 - Relationship Mapping

Lesson Preparation
None.

What to Watch For
None.

Connections
Remind them that the barred relationships to ORDER LINE indicate that the UID of ORDER and FOOD ITEM participate in the UID of ORDER LINE. When the entity is mapped to table ORDER_LINES, we see that the foreign keys from FOOD_ITEMS and ORDERS (fim_number and odr_number, respectively) make up the composite primary key of ORDER_LINES.
What Will I Learn?

In this lesson, you will learn to:

- Apply the rule of relationship mapping to correctly transform 1:M and barred relationships
- Apply the rule of relationship mapping to correctly transform M:M relationships
- Apply the rule of relationship mapping to correctly transform 1:1 relationships
- Apply the rule of relationship mapping to correctly transform relationships in an arc
Why Learn It?

What if you were building a house for someone? You bought all the materials — wood, paint, doors, windows, nails, screws, etc. — but didn’t have information about how they all came together? You didn’t know how many rooms there were, where the windows were supposed to be, where the doors faced, or what color each room was. You could build a house, but it might not be the house that the customer had in mind.

Relationships are mapped to foreign keys that allow tables to reference each other. Foreign keys enable users to access related information from other tables. If we don’t map relationships, we just have a lot of stand-alone tables containing information that can’t be related to the rest of the database.

Mapping relationships to relational database structures is part of creating the “first cut” database design that will serve as the basis for further discussion among designers, developers, and database administrators.
Tell Me / Show Me

When transforming M:1 relationships, it may be helpful to visualize the crow’s foot “pulling” the UID of the entity on the one side into the entity on the other side (the crow’s feet side). Point out that the ER diagram captures the relationships between entities, expressed in business terms. When the conceptual model is transformed, the relationships become foreign-key columns, but the relationship name itself is not carried over. The database design will be the basis for the system, but starting with a conceptual model ensures that the tables, columns, and constraints created in the database are relevant to the business and fulfill its requirements. A database contains objects such as tables, columns, and constraints. A data modeler sees how these objects are related from a business perspective.

“In the example, dpt_id is mandatory, and epe_id is optional.” Give the business example so that students understand why the epe_id is optional. Does the president of a company have a manager?

Since epe_id implements the recursive relationship in EMPLOYEE, it is probably better renamed to mgr_id. Decisions like this can be made at this stage of mapping.
The program will have to check for two things:
- You cannot create a master record without at least one detail record. (In the example, BAND is the master and MUSICIAN is the detail.)
- When deleting details, you must be sure that you do not delete the last detail for a master record, or alternatively, you must delete the master record together with the last detail.
In the example, a paycheck may not be transferred to another employee. This means that the epe_id, which is the foreign-key column in the table PAYCHECKS, cannot be updated.
Tell Me / Show Me

Help students visualize the data in ACCOUNTS. The act_number alone would not be unique within the table, but the combination of act_nbr and bak_nbr would be.
The UIDs and resulting primary keys are highlighted in different colors to help you trace a UID through the hierarchy.

The primary key of FLOORS now becomes a composite of flr_nbr and bldg_id. The primary key of SUITES is a composite of sue_nbr, flr_nbr, and flr_bldg_id. Point out that the composite is one primary key (a table is only allowed to have one PK).

In this case, it is also one foreign key for each table, even if that key is a composite of multiple columns. The foreign key in SUITES is the combination of flr_nbr and flr_bldg_id. The foreign key in ROOMs is a combination of sue_nbr, sue_flr_nbr, and sue_bdg_id.
Tell Me / Show Me

Ask students: What is the primary key of ROOMS?
Answer: The combination of rom_nbr, sue_nbr, sue_flr_nbr, and sue_bdg_id.
Verify that the combination of all four is what makes a row unique. Point out that any three columns alone or in combination does not make a unique value.
Trace the id all the way from BUILDINGS (id) to FLOORS (bdg_id) to SUITES (flg_bdg_id) to ROOMS (sue_bdg_id).
Tell Me / Show Me

Note the subscript notation on each foreign key (fk1 and fk2). This is meant to show that each foreign-key column references the primary key of a different table. This differs from previous examples where the foreign-key column (or columns) references the primary key of a single table.
Tell Me / Show Me

Clarify the necessity for a unique key on bcp_code if necessary. Remind them of the relationship in the conceptual model: “Each bottle cap may be the sealer for one and only one soda bottle.” If you had a bottle cap with code AB55, that would be unique in the SODA_BOTTLES table, because it can occur only in one instance or one row of the table.
Tell Me / Show Me

Mandatory 1:1 relationships are rare. In most cases, this will be modeled as a single table, without a need for a 1:1 relationship. However, if you need to clarify the limitation of 1:1 mandatory at both ends, you can revisit the earlier slide on m:1 mandatory on the “one” end in this lesson.
Tell Me / Show Me

Point out that pse_id and phe_id in the EVENTS table are optional, but the relationships in the arc are mandatory (each EVENT must be held at one VENUE). A check constraint will enforce this mandatory relationship.
Tell Me / Show Me

You may need to explain the concept of a check constraint, as opposed to additional programming that is needed to enforce mandatory relationships on the “one” end (as in the previously discussed M:1 or 1:1). A check constraint is programming code that can be stored in the database. It can enforce simple rules that apply to a single row in the table (such as comparing values or ensuring that they are null or not null). This is the case with the arc.

In the case of a mandatory one end of a M:1 or 1:1 relationship, we have to check that if a row is entered in one table (the master), a row must also be entered in another table (the child, or detail). A check constraint cannot span two tables or different rows in the same table. It cannot prevent insert, update, or delete operations. This is why additional programming (instead of a check constraint) is necessary.
We have not mapped the STAFF entity yet. Tell students to use stf_id as the pk for STAFF. They will need it for the ORDERS table.

Also, ask students why it makes sense to transform ORDER LINE last (after FOOD ITEM and ORDER). Answer: Because ORDER LINE is on the many end of 1:M relationships with FOOD ITEM and ORDER. This means that you will need the primary keys of the two tables at the one end, because they will become foreign keys in ORDER_LINES.

You may also want to add that as a general rule for any ERD, they should map the entities on the one end of M:1 relationships first.

You can have the students refer to activity in Section 12, Lesson 2, for the table definitions of SHIFTS, REGULAR MENUS, PROMOTIONAL MENUS, and FREQUENT DINER CARDS.
A check constraint is needed to enforce: either (rmu_code is null and pmu_code is not null) OR (rmu_code is null and pmu_code is not null).

Additional programming is needed to ensure that stf_id is not updateable. This will enforce the non-transferability of the relationship between ORDER and STAFF.
Lesson 4 - Subtype Mapping

Lesson Preparation
You may want to review supertypes and subtypes briefly. Use the DJ and Global Foods ERDs as examples.
You may want to review supertypes and subtypes briefly. Use the DJ and Global Foods ERDs as examples.

**Global Fast Foods**

Students may get bogged down in the arc implementation of a supertype. The best way to understand it is to redraw the data model using an arc instead of a supertype. Then transform the model as an arc. The tricky part is that the relationships in the arc are 1:1 instead of M:1, which is usually seen in an arc.

1:1 relationships are not modeled that often; usually the data modeler will use one entity instead of two. For this reason, modeling supertypes as arcs is not that common either. Arcs are best used to express exclusive relationships, not a “type of” condition.

Let students know that most of the time, the single-table implementation is used.
Connections

Section 12, Lesson 3, also discusses mapping 1:1 relationships, as well as arcs. Students may need to review this to understand why a 1:1 relationship results in a unique key on the mandatory side.

For a review of supertype modeling, refer to Section 4, Lesson 3.

For a review of arcs and exclusive relationship modeling, refer to Section 7, Lesson 3.
What Will I Learn?

Tell students that there are three ways to implement supertypes in the database. All three will be discussed in this lesson.
Why Learn It?

A carpenter who is building your dream house may know that you will use different types of light bulbs all around the house. However, if you do not provide information on where certain types of light bulbs should be installed, you could end up with an overly bright bedroom and a dimly lit kitchen.

Mapping supertypes and subtypes makes sure that the right information gets stored with each type.
Point out that salary and hourly_rate are both optional, even if the ERD has them as mandatory. Dpt_id can remain mandatory, because it comes from the relationship to the supertype EMPLOYEE. Agy_id comes from a relationship to the subtype and has to become optional.

Epe_type is the discriminator column. It can have values of FTE or PTE. Notice that mgr_id represents the recursive relationship for EMPLOYEE. It is a foreign key column and would normally be named epe_id, after the parent table. However, it makes sense to rename it at this stage, to make it easier to understand.
Tell Me / Show Me

A check constraint is necessary to ensure that (epe_type = ‘FTE’ and salary is not null and hourly_rate is null and agy_id is null) OR (epe_type ‘PTE’ and salary is null and hourly_rate is not null and agy_id is not null).

Epe_id is the employee identifier of each employee’s manager.

A note about the OTHER subtype:

An OTHER subtype is recommended in the conceptual model to ensure that the subtypes are exhaustive. However, by the time we start the design phase, we should have done extensive analysis to determine if another subtype is truly needed. If so, then this subtype must be named, and its attributes specified. If not, then the OTHER subtype is not part of the transformation process.
Tell Me / Show Me

**Section 12**

---

**Lesson 4 - Subtype Mapping**

---

**Tell Me / Show Me**

---

**Supertype Implementation: Single Table (continued)**

In the ER model, salary is mandatory for full-time employees, and hourly rate is mandatory for part-time employees. When the EMPLOYEE supertype is implemented as a single table in the relational model, these attributes become optional. A check constraint is needed to enforce the business rule modeled in the ERD.

In the example, the code for the check constraint would look like this:

```sql
CHECK (epe_type = 'FTE' and salary is not null and hourly_rate is not null and agy_id is not null)
    OR (epe_type = 'PTE' and salary is null and hourly_rate is null and agy_id is null)
```

The code checks that if it is a full-time employee (epe_type = 'FTE'), then there must be a value in the salary column and the hourly_rate and agy_id columns must be empty. Conversely, if it is a part-time employee (epe_type = 'PTE'), then there must be a value in hourly_rate, and an agy_id, but salary must be left blank.

---

**Tell Me / Show Me**

Explain that a check constraint is activated when you insert or update a row in the table. So, if you tried to update the row for Marcus Rivera by putting a value in the salary column, the check constraint would prevent you from succeeding. Similarly, if you wanted to enter a new row for a full-time employee (epe_type = FTE), but tried to enter a value for agy_id, the check constraint would prevent the insert.
Tell Me / Show Me

This is the implementation used most often. It is elegant and appropriate for most situations and should be considered as the first choice.

Usually you would create a view for every subtype, showing only the columns that belong to that particular subtype. The correct rows are selected using a condition based on the discriminator column. Example:

```
CREATE OR REPLACE FULL_TIME_VW AS
Select id, first_name, last_name, salary, dpt_id, epe_id from EMPLOYEES where epe_type = 'FTE';
```
Tell Me / Show Me

Subtype Implementation: Two Table

This is also called "two-table implementation." You create a table for each of the subtypes. So, in reality, you could have more than two tables, if you had more than two subtypes.

Rules:
1. Tables: One table per first-level subtype.
2. Columns: Each table gets a column for all attributes of the supertype with the original optionality. Each table also gets a column for each attribute belonging to the subtype, also with the original optionality.
3. Identifiers: The primary UID at the supertype level creates a primary key for each table. Secondary UIDs of the supertype become unique keys in each table.
4. Relationships: All tables get a foreign key for a relationship at the supertype level, with the original optionality. For relationships at the subtype level, the foreign key is implemented in the table it is mapped to. Original optionality is retained.

Tell Me / Show Me

Point out that the original optionality of the attributes and relationships is carried over to the columns and foreign keys, so there is no need for a check constraint as there is in the one-table implementation.

If there were a need to have unique values of the primary key (ID) across both tables, then additional programming would be necessary.
Tell Me / Show Me

Point out that id, material, and mnr_id come from attributes in OR relationships to the supertype. Therefore, they appear in both tables.
**Tell Me / Show Me**

**Usually you would create an additional view that represents the supertype, showing all columns of the supertype and the various subtypes. The view SELECT statement must use the UNION operator.**
Tell Me / Show Me

We have previously discussed how supertypes and subtypes can also be drawn as an arc. We are illustrating this with the example, so that students can better understand the arc implementation.
Tell Me / Show Me

Discuss why the relationships are 1:1 mandatory and exclusive:
- Each SHIRT must be one piece of CLOTHING. Each SHOE must be one piece of CLOTHING.
- Conversely, each piece of CLOTHING must be either a SHOE or a SHIRT but not both.

Point out that most of the time, the data modeler would choose to model this with a supertype (previous diagram) because of the 1:1 mandatory nature of the relationships. Recall that 1:1 mandatory relationships usually indicate that a single entity is needed. Arcs are more often used when the exclusive relationships are 1:M.
Tell Me / Show Me

Point out that it is not necessary to remodel the supertype as an arc (as we did), in order to choose to do an arc implementation. However, it makes sense to do so for clarity.

If necessary, recall the previous lesson on mapping an arc. You would need a check constraint on CLOTHING to enforce that (sht_id is not null and she_id is null) OR (sht_id is null and she_id is not null).
Although you would hardly use them, you could consider creating additional views that represent the supertype and subtypes in full.
Assessment:
After going through the activity, you may want to conduct the following discussion to check for understanding:

- Remind students that they need a discriminator column. What would this column be named? (Answer: pnr_type.) What are the allowable values for this column? (Answer: EPR, DJ, MNR.)
- Remind students that they need a check constraint. What would it check for? (Answer: (pnr_type = ‘EPR’ and expertise is not null and specialty is null and authorized_expense_limit is null) or (pnr_type = ‘DJ’ and expertise is null and specialty is not null and authorized_expense_limit is null) or (pnr_type =’MNR’ and expertise is null and specialty is null and authorized_expense_limit is not null).)
- What does the foreign-key column pnr_id refer to? (Answer: the PARTNERS table.)
- What relationship was it mapped from? (Answer: the recursive relationship in PARTNER.)
- What does it represent? (Answer: the id of the partner’s manager.)
<table>
<thead>
<tr>
<th>Key Type</th>
<th>Optionality</th>
<th>Column Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>pk</td>
<td>*</td>
<td>id</td>
</tr>
<tr>
<td></td>
<td>*</td>
<td>first_name</td>
</tr>
<tr>
<td></td>
<td>*</td>
<td>last_name</td>
</tr>
<tr>
<td></td>
<td>*</td>
<td>expertise</td>
</tr>
<tr>
<td></td>
<td>*</td>
<td>specialty</td>
</tr>
<tr>
<td></td>
<td>*</td>
<td>authorized_expense_limit</td>
</tr>
<tr>
<td>fk</td>
<td>*</td>
<td>pnr_id</td>
</tr>
<tr>
<td></td>
<td>*</td>
<td>pnr_type</td>
</tr>
</tbody>
</table>
Try It / Solve It

Assessment:
After going through the activity, you may want to conduct the following discussion to check for understanding:
- What does the foreign-key column mnr_id refer to? (Answer: the MANAGERS table.)
- What relationship was it mapped from? (Answer: the recursive relationship in STAFF.)
- What does it represent? (Answer: the id of the manager.)
### COOKS (COK)

<table>
<thead>
<tr>
<th>Key Type</th>
<th>Optionality</th>
<th>Column Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>pk</td>
<td>*</td>
<td>id</td>
</tr>
<tr>
<td></td>
<td>*</td>
<td>first_name</td>
</tr>
<tr>
<td></td>
<td>*</td>
<td>last_name</td>
</tr>
<tr>
<td></td>
<td>*</td>
<td>date_of_birth</td>
</tr>
<tr>
<td></td>
<td>*</td>
<td>salary</td>
</tr>
<tr>
<td></td>
<td>*</td>
<td>training</td>
</tr>
<tr>
<td>fk</td>
<td>*</td>
<td>mnr_id</td>
</tr>
</tbody>
</table>

### ORDER_TAKERS (OTR)

<table>
<thead>
<tr>
<th>Key Type</th>
<th>Optionality</th>
<th>Column Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>pk</td>
<td>*</td>
<td>id</td>
</tr>
<tr>
<td></td>
<td>*</td>
<td>first_name</td>
</tr>
<tr>
<td></td>
<td>*</td>
<td>last_name</td>
</tr>
<tr>
<td></td>
<td>*</td>
<td>date_of_birth</td>
</tr>
<tr>
<td></td>
<td>*</td>
<td>salary</td>
</tr>
<tr>
<td></td>
<td>*</td>
<td>overtime_rate</td>
</tr>
<tr>
<td>fk</td>
<td>*</td>
<td>mnr_id</td>
</tr>
</tbody>
</table>

### MANAGERS (MNR)

<table>
<thead>
<tr>
<th>Key Type</th>
<th>Optionality</th>
<th>Column Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>pk</td>
<td>*</td>
<td>id</td>
</tr>
<tr>
<td></td>
<td>*</td>
<td>first_name</td>
</tr>
<tr>
<td></td>
<td>*</td>
<td>last_name</td>
</tr>
<tr>
<td></td>
<td>*</td>
<td>date_of_birth</td>
</tr>
<tr>
<td></td>
<td>*</td>
<td>salary</td>
</tr>
<tr>
<td></td>
<td>*</td>
<td>budget</td>
</tr>
<tr>
<td></td>
<td>*</td>
<td>target_revenue</td>
</tr>
<tr>
<td>fk</td>
<td>*</td>
<td>mnr_id</td>
</tr>
</tbody>
</table>
Lesson Preparation

For this lesson, we will perform the Try It/Solve It activity (use HTML DB) before we explain what happened in Tell Me/Show Me.

Oracle HTML DB User/Student Guide

This document will help you become familiar with how to use Oracle HTML DB and each of its components from and end user’s perspective. If you would like to learn more about the administrative capabilities or teacher capabilities of HTML DB, please refer to the Academy HTML DB Teacher Guide.

To log into HTML DB, go to http://iacademy.oracle.com

Enter your school, username, and password information. This information was given to you via email from academy_us@oracle.com. If you did not receive this information, please email: academy_us@oracle.com. You must be a registered and paid DM/SQL instructor. Students will get their account information from their teacher.

The username is broken down into four parts: 1. country (US) state (VA); 2. school-naming convention given to you by Academy (XYZ); 3. the SQL section (SQL01) -- if you teach more than one class, you may have students in SQL02, SQL03, etc.; and 4. the student number (S01).
Once you log into HTML DB, you will see the Main page that consists of all the components of SQL Workshop and Data Browser. These are the main components to use with the DM/SQL curriculum until the SQL final project using Application Builder. You can also click the tabs at the top of the page for quicker access to frequently used components.

SQL Workshop: This is where you will be running all of your SQL statements.

Data Browser: Allows you to view database objects such as tables, views, indexes, etc.

Using SQL Workshop: Ad Hoc SQL

The Ad Hoc SQL (also known as the SQL Command Processor) window is where you will be spending most of your time entering and practicing SQL code.

There are a few things to note about the SQL Command Processor window:

1. Students have access only to their own schemas.
2. You can select to have the output of your command displayed in HTML or in a Microsoft Excel spreadsheet.
3. Above the window where you enter your commands, note that it says "Enter a SQL or PL/SQL Statement." You will be learning only SQL in your class. PL/SQL (Procedural Language SQL) is Oracle’s extension to the SQL language. It will not be taught in your class.
4. You have the ability to run your SQL statement or save it for future use. Saving your SQL statements helps when you are learning so that you do not have to type the statement multiple times. The saved SQL statement can be found in the Archive section.

After the SQL statement is executed, the results are displayed.

You can increase the number of rows returned by selecting Max Rows from the drop-down box. This will help in reducing the number of pages that you need to flip through to see all of the rows returned.

Saving a SQL statement from the Command Processor Window:

To save a SQL statement, simply click Save.

Once you have clicked Save, you will be prompted to give your file a name. You can provide a description of the file if you prefer, but this is not necessary in saving your statement and will not appear in the Archive section.

Click Save to save your statement.

Notice that the statement (script) was successfully saved. There are a few items to point our when saving your script.

1. The statement is now saved in the Archive section. To access your archived files, click the Archive link.
2. There is a Find field to search on a particular file. This is helpful if your Archive folder gets large.
3. To run your statement from the Archive page, simply click the Run link under the Action column.
4. To remove a file from the Archive page, select the file by clicking the box to the left of the filename. Once you click the check box, click Delete Checked.
What to Watch For

Students will quickly catch on to HTML DB and creating tables. Encourage them but set a standard for appropriateness.

Connections

None.
What Will I Learn?

In this lesson, you will learn:

- Log into HTML DB
- Create the music table using the script provided
- Enter some sample data into the music table created in this lesson
- Query the music table to view the new data
- Refine the query
Why Learn It?

Why Learn It?

You have drawn the plans for your dream house and then transformed it into a physical design for the house. The next step is to actually build the house. You will need tools to do this.

HTML DB is a tool that will allow you to build tables in an Oracle database.
Tell Me / Show Me

Review the HTML DB User's Guide. Step through it with students, making sure each student can log on successfully.
Try It / Solve It

You will be creating a database table called MUSIC. The table data is provided here. Can you identify the column names? How many rows does the table have? What is the unique identifier of this table? Can you give an example of an instance of the MUSIC table?

<table>
<thead>
<tr>
<th>MUSICID</th>
<th>ARTIST_NAME</th>
<th>TYPE</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>BOB MARLEY</td>
<td>REGGAE</td>
</tr>
<tr>
<td>2</td>
<td>BRITTNEY SPEARS</td>
<td>POP</td>
</tr>
<tr>
<td>3</td>
<td>NSYNC</td>
<td>POP</td>
</tr>
<tr>
<td>4</td>
<td>SHAGGY</td>
<td>REGGAE</td>
</tr>
<tr>
<td>5</td>
<td>JIMMY CLIFF</td>
<td>REGGAE</td>
</tr>
<tr>
<td>6</td>
<td>KID ROCK</td>
<td>ROCK</td>
</tr>
<tr>
<td>7</td>
<td>THIRD WORLD</td>
<td>REGGAE</td>
</tr>
<tr>
<td>8</td>
<td>LENNY KRAVITZ</td>
<td>ROCK</td>
</tr>
<tr>
<td>9</td>
<td>YELLOW MAN</td>
<td>REGGAE</td>
</tr>
</tbody>
</table>

Try It / Solve It

Give each student a copy of the MUSIC table.
Have them answer the questions ABOVE.

**Answers:** Column names are musicid, artist_name, type. There are nine rows in the table. The unique identifier is musicid. Any row in the table can be an example of an instance.
Try It / Solve It

Students can either enter the music-table statements themselves in HTML DB or use the music.sql script with the statements already written. The CREATE TABLE statement must be run first and then each INSERT statement entered one at a time.

Explain that SQL stands for structured query language, and they are going to use SQL commands to create a table in the database.

1. Give each student a copy of the music.sql file.
2. Walk them through the SQL syntax: CREATE TABLE <table name> is the SQL command. In the example, the table name is MUSIC. This command must be followed with a list of columns and their data type and any constraints. This list must be enclosed in parentheses. The most common data types are varchar(length), number(length), and date. In the example: MUSICID is the first column, with data type NUMBER(2); ARTIST_NAME is the second column with data type varchar20). NOT NULL specified after a column name and data type means that the column is mandatory. In the example: MUSICID is mandatory. After the list of columns, you can also specify a constraint for the table, with the word CONSTRAINT. In the example: the constraint is a primary key on the column MUSICID. Each SQL statement ends with a semicolon.
3. Log on to HTML DB. Wait until all students have successfully logged on.
4. Step students through the process for entering the CREATE TABLE statement in the SQL window.
   If they prefer to run the script, instruct them to use the command @music.sql.
5. HTML DB should return a "table created successfully" statement. Don't continue until all students have successfully created the MUSIC table.
Try It / Solve It

Explain to students that the INSERT statements must be entered and run one at a time.
Explain to students that they used HTML DB to log on to the Oracle database and ran a SQL script (or entered SQL statements) to create a table and enter data into the database.