**Oracle Database 12c**

1. Oracle Database 12c provides support for the \_\_\_\_\_ and \_\_\_\_\_ database models.

Relational, object relational

1. Oracle SQL is a procedural query language.

False

1. The acronym DDL stands for which type of Oracle SQL command?

Data Definition Language

1. The alternative to quoting strings with single quotes is double quotes.

False

1. What single-row function would be most appropriate to use to convert a DATE column to the format "YYYY/MM/DD HH:MI:SS" for a report?

To\_Char not To\_DateTime

1. You want to roll up totals by department, but only see the totals for two of the departments. To do this, you'll have to use the \_\_\_\_\_ and the \_\_\_\_\_ clauses.

Group by and Having not Group by and Where clauses

1. Your query using set operators has multiple subqueries and you want to change the order of evaluation. What is the best method for changing the evaluation order?

Put parentheses around pairs of queries that you want to run before any other set operators.

1. You're using SQL Developer and running some INSERT and UPDATE statements. What is the easiest way to perform a COMMIT on your work so far?

Click on Commit button on the top of the SQL window

1. A datafile in a \_\_\_\_\_ tablespace is limited to about 64 GB if you have a 16K block size.

SMALLFILE not BIGFILE

1. The \_\_\_\_\_ logical database structure is the most granular in an Oracle Database.

Data block

1. What is the difference between a view starting with V$ and a view starting with GV$?

The GV$ view contains an INST\_ID column that identifies which database instance in the cluster this row is referencing.

1. Blocks storing a(n) \_\_\_\_\_ column does not have nor need a setting for PCTFREE.

CLOB not XML not LONG and not IOT

1. Both \_\_\_\_\_ and \_\_\_\_\_ need data in the UNDO tablespace.

Queries and DML

How to install Oracle 12 c:

First install jdk file from Oracle website

<https://www.oracle.com/technetwork/java/javase/downloads/jdk10-downloads-4416644.html>

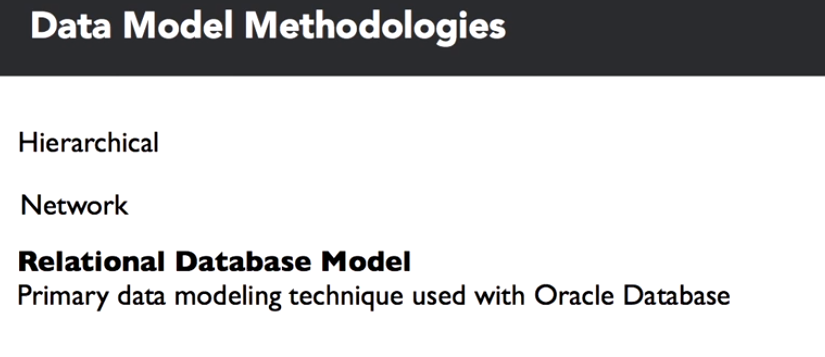
Install SQL Developer from <https://goo.gl/nftWBA> -> Redirects to

<https://www.oracle.com/technetwork/developer-tools/sql-developer/downloads/index.html>

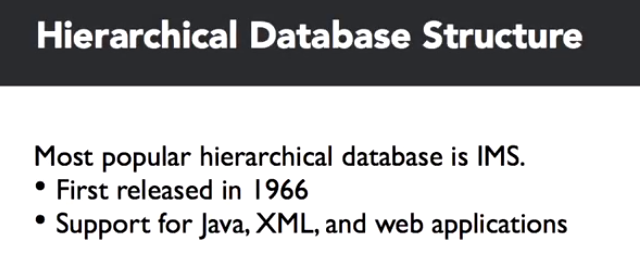
what you should know:

* Access to an Oracle 12 c database
* Install SQL Developer (free) -link above
* Basic understanding of relational databases
* Some experience with a dialect of SQL
* Understanding of client-server architecture and networks
* Database application development experience helpful

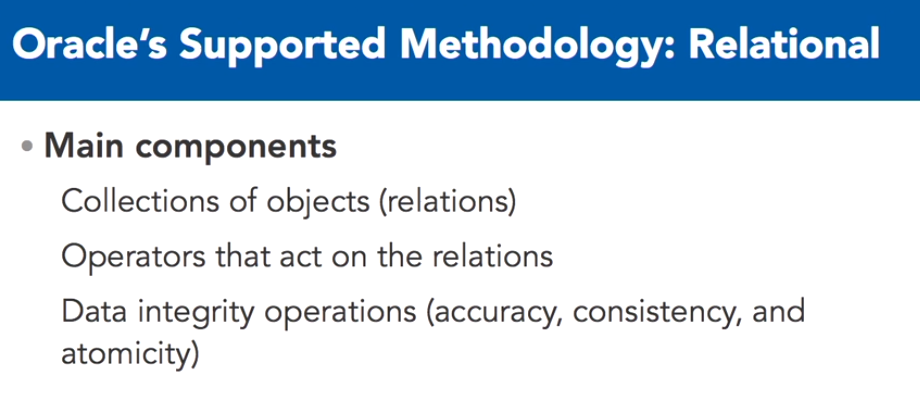
Whether you're using a relational database, hierarchical, network, or an object-relational database, you need to use a **data modeling tool** to document the objects, attributes and relationships between the objects in the database. In the '60s and '70s, both hierarchical and network data models were more prevalent, mostly because of limitations with the hardware and software available at the time. As computing power and capabilities increased, other models emerged such as the relational database model. The primary data modeling technique used with Oracle Database is the relational model.



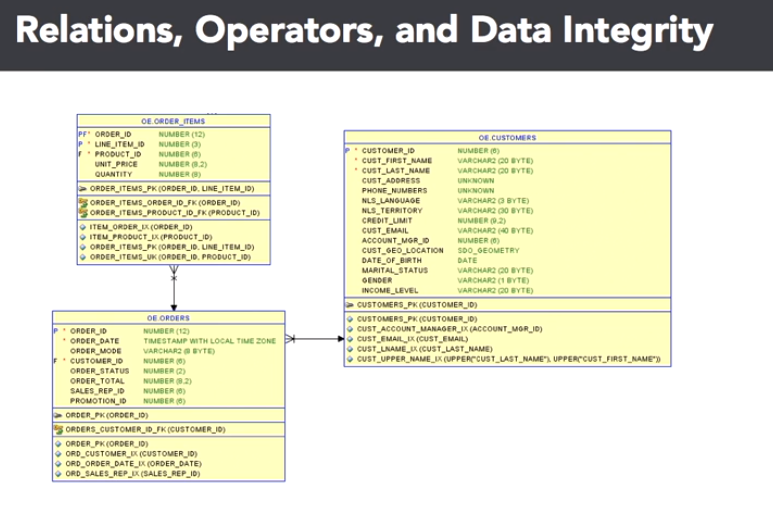
The most popular hierarchical database, IMS, was first released in 1966 by IBM in conjunction with Rockwell and Caterpillar as a database to support bill of materials, application management for the Apollo space program. IMS has evolved enough to be a viable platform well into the 21st century with support for Java, XML and web applications. IDMS, Integrated Database Management System, is an example of the network database model, first developed at BF Goodrich in 1973 and later marketed commercially by Cullinet.



Relational database modeling was born in the early 1970's with a paper by Dr. E. F. Codd in June of 1970, entitled, "A Relational Model of Data for Large Shared Data Banks." So, what is a relational model? It's really three things: a collection of objects, relations, operators that act on the relations, and data integrity operations.

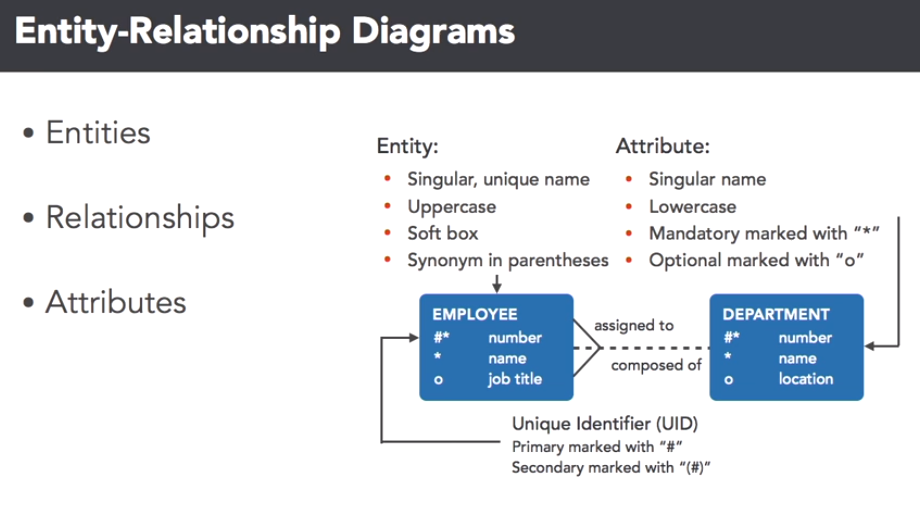


The collection of objects represent real world things such as a list of auto parts, a balance sheet, or in this case, customer orders, much like a spreadsheet.

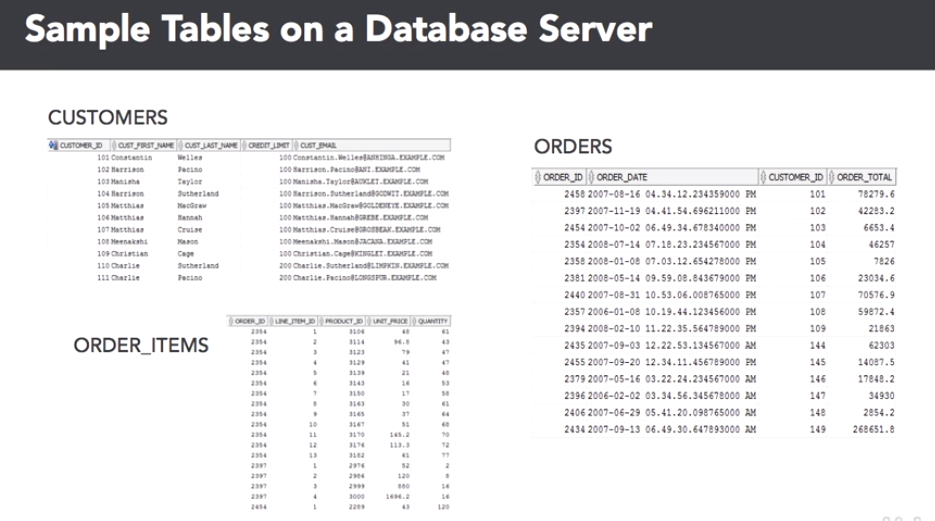


These are what are called relations. Second, a relational model has a set of operators that act on one or more relations. These operators include filtering the components of a relation to get the desired results. In a spreadsheet, the original version of a database table, you might use a filter operation to only view certain columns, or you might only want to see certain rows of the spreadsheet. Finally, you must have one of the most important components of the relational model: data integrity operations.

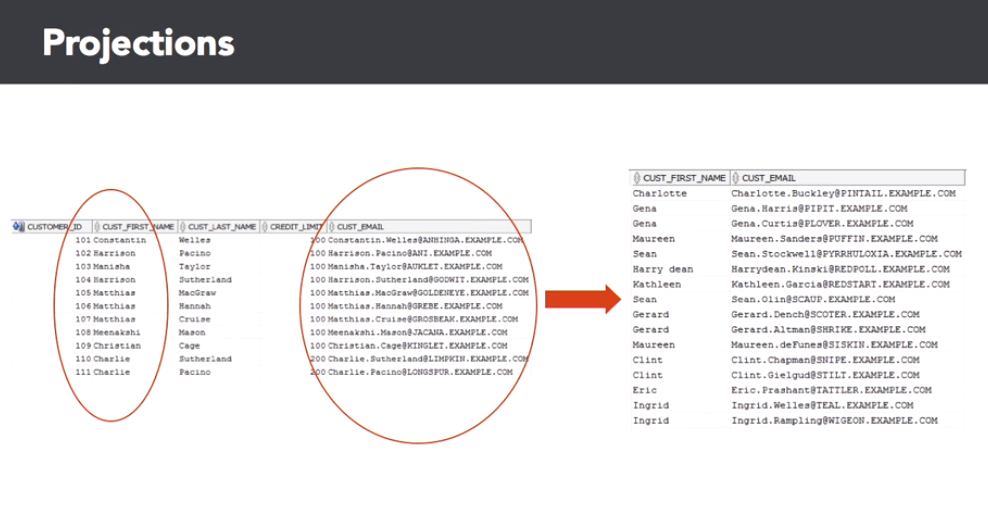
For subcomponents of any given relation, you want to make sure that the information in the relation is accurate and logically consistent with the information in the other relations of your data model. We'll point out these components in the following slides. You've probably noticed some interesting nomenclature and symbols in the previous slide. That is the logical implementation of those business entities in Oracle Sequel Data Modeler, of an entity relationship diagram. These diagrams make it easy to identify relations among the entities along with the business rules that apply to the attributes in each entity.



We'll use this notation throughout this course when referencing logical data model components. Here, we see a small subset of tables in the database.

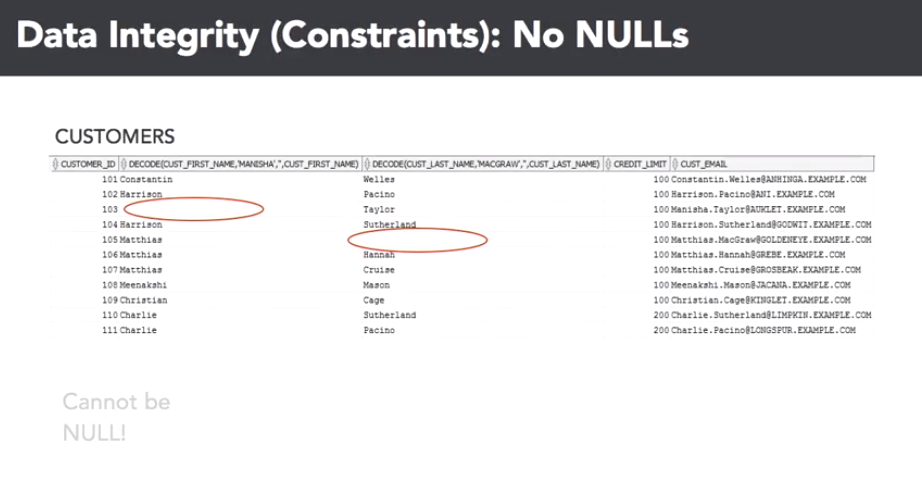


We'll get back to the data modeling development part in a bit. The three relations stored as tables in the database are the customers, orders and order item tables. Notice that there are many operators you can use on these tables. For example, you may use a projection on the customer's table to only see a customer's first name and e-mail address or you may want to see all information about a single customer's orders using a selection on the order items table.

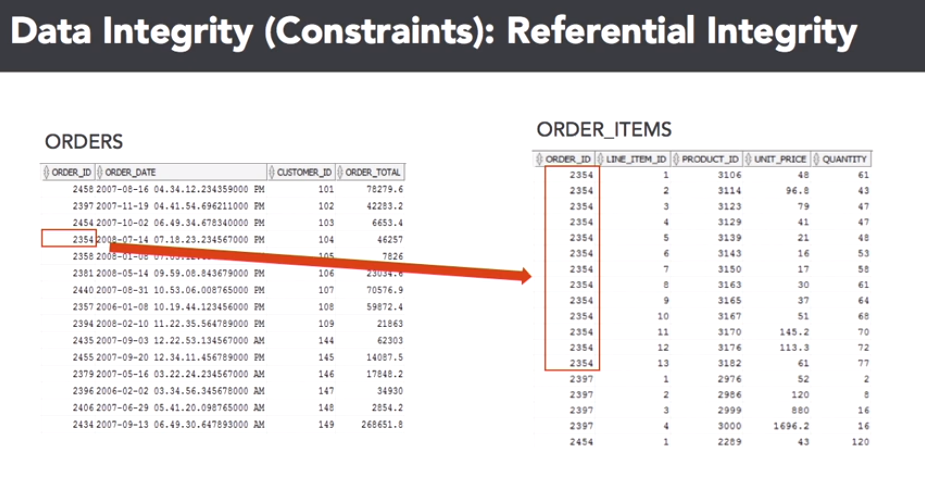


Finally, you may want to use a join operator on the orders and order items table to generate a new relation that shows a given order number and its associated line items.

Let's talk about data integrity next. When defining a relation, you may also want to define the data integrity operations that ensure data quality and consistency. For example, you don't want to have any missing data for custfirst name or custlast name in the customers table, as you can see in the slide.



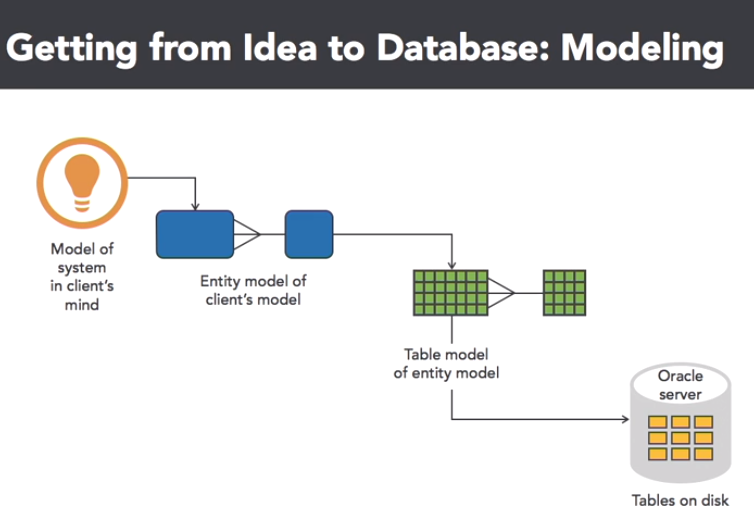
Also, you want to make sure that the order ID column in the order items table has an entry in the orders table. This is called a foreign key constraint.



How do you get from an idea in a CEO's head to the actual data model? Well, when you want to create an Oracle Database for an application to fulfill a business need, you usually don't want to start creating tables on a new database as your first step. It's important to start with the application stakeholders to gather business requirements for the data model. Frequently, the application owners have an idea in their head of the types of data they want to store, but that doesn't translate directly to Oracle Database tables.

After the business requirements are gathered, you'll facilitate the creation of an entity relationship, ER diagram, to capture the entities, relations and their attributes, which are usually columns, that are important to the application along with the relationships between the entities and the data integrity constraints. This would be considered the logical ER diagram. From the logical ER diagram, the modeler will generate the database table model for the diagram, in other words, the Oracle Database statements that will be executed to create the physical representations of the entities, attributes and relationships, not to be confused with relations, defined in the ER diagram.

Finally, the Oracle DBA, which may also be you, will run the commands to create the first version of the database tables for the application.

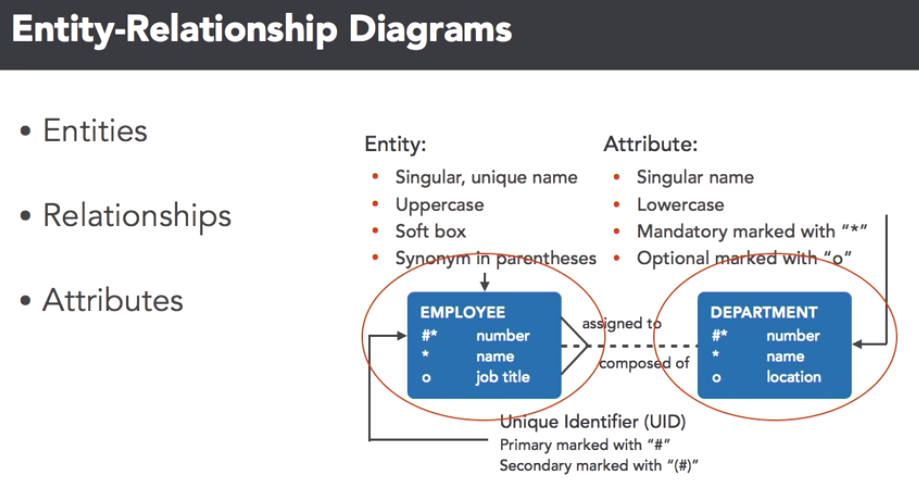


So, what do we know now? Several things. Oracle Database is primarily a relational database platform with some object-oriented capabilities thrown in, and is firmly based on the relational database model, first proposed by E. F. Codd in 1970. An entity relationship ER diagram shows the logical objects and relationships in a relational database.

These objects will map almost one to one with the physical objects in the database itself.

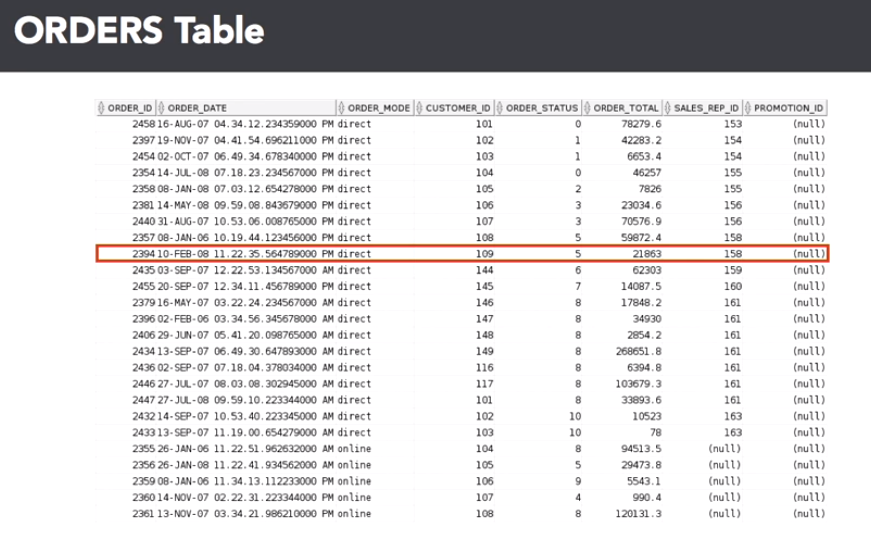
Logical database modeling

Let's start out this lesson by analyzing the components in a logical data model diagram.



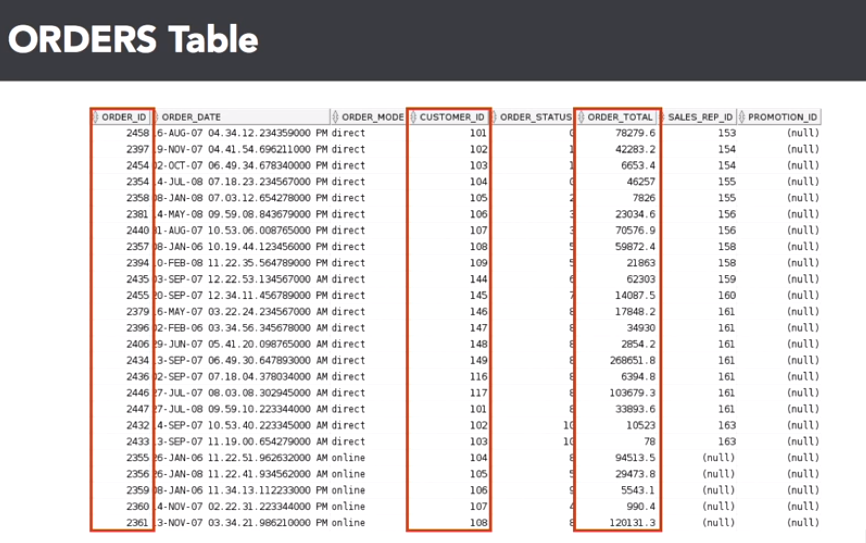
At this level, you're not as concerned about data types, but instead, what elements of your organization are important from a business perspective. Each blue box is one of the entities, remember, this is also called a relation and data modeling terminology, in this case, employees and departments. You'll use certain conventions in your data modeling tool, to assign characteristics to the attributes of each entity.

The entity name is in upper case, the attributes are in lower case, and both are almost always singular. You'll also want to mark certain attributes as being unique, or not knowable, to enforce data integrity. Here, the employee number, and department number, are unique identifiers, and the name attribute in both of those entities, can't be empty. No worries, a lot of these requirements are easy to implement, in most data modeling tools, as you'll see later in this video. In your favorite data modeling tool, you'll also define the relationships between the entities.

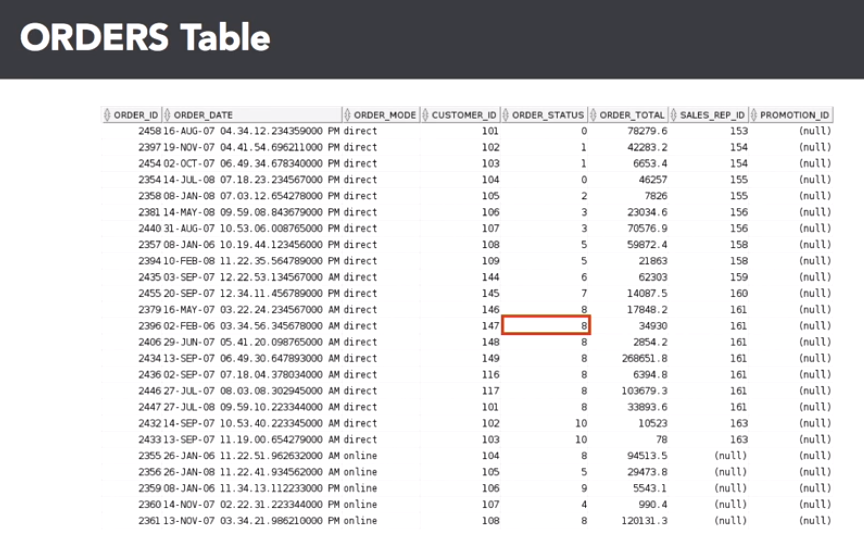


This is a critical step to ensure that data integrity will be enforced using database features such as foreign keys, as well as defining the cardinality of the entities in your database. After you've created the table in the Oracle database, it will look something like this in SQL Developer. Let's review the connection between the elements in your data model diagram, with the database table itself. Here are the first 25 rows of the order table. A row in the database, is a tuple in a classic relational database model.

And you select rows using relational database languages, such as a SQL SELECT statement structured query language. In this diagram, the row with an order ID of 2394 is highlighted. A column, also known as a **projection**, is the attribute of an entity. A column can have several attributes beyond what kind of data type it is. It can be, one, a unique identifier, two, an identifier that links to a unique column in another table.

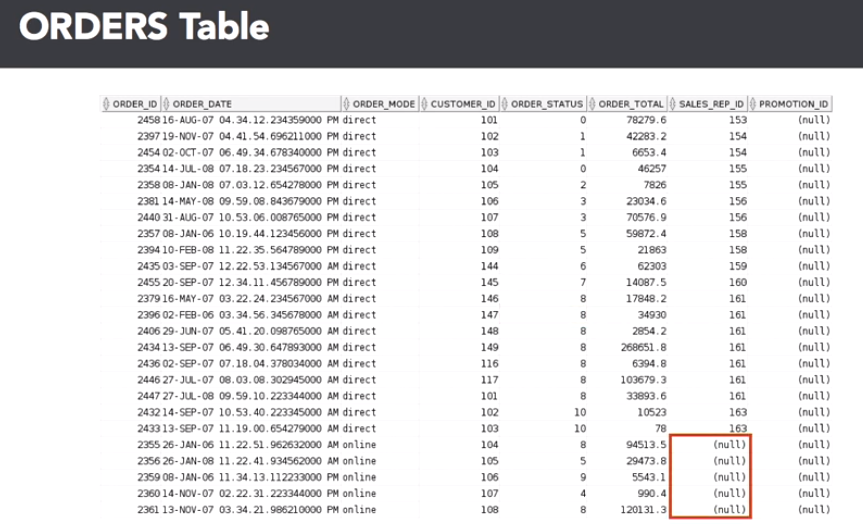


Or three, just to another column, usually dependent on the unique identifier in that row. An intersection of a row and column is a field, a single atomic value of a column, in a single database row.

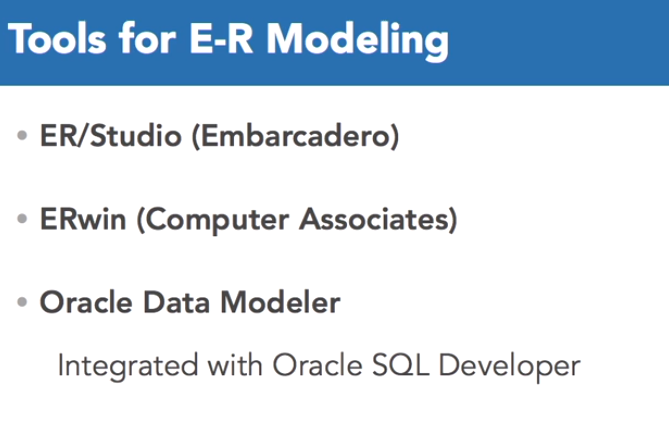


The highlighted field answers the question, "What is the order status for order number 2396?" For a customer service representative that is inquiring on the order status, seeing a value of eight might not be very useful or obvious, therefore the result of this query will probably be joined to the order status table, to get the text description.

Finally, any non-primary key columns can have a value of null, meaning that the field has no value, or it's unknown. It does not mean that it is blank, or has spaces, or has a value of zero, it does not exist. This has many implications when selecting rows, or joining tables, based on the value, or lack thereof, in this column.



Now that you know what you want to do, which is to create the data model, you can use one of many good tools from vendors such as Embarcadero, and Computer Associates, to perform your data modeling tasks.



A big advantage to using Oracle Data Modeler, is that it's integrated into Oracle SQL Developer, allowing you to manage the entire database development and deployment life cycle within one gooey, and on top of it all, it's free.

Here's how well-integrated Oracle SQL Developer is with Oracle Data Modeler. Going to the file tab, select data modeler, then select import. As you can see, there are a wide variety of other data sources, in this case we're going to import from data dictionary.

Next, pick a connection to the database you want to use for the import, I'm going to click on the OE connection. Click on next, and you'll see a list of the schemas in the database. I'm going to pick the OE schema, since it contains the order entry tables. Go ahead and click that check box, and then click next. Right here, you'll see all the tables that OE owns, and there are the two we want, orders, and order items. So I'm going to select both of those, and then click on next.

Before performing the import, you'll see a summary screen, click on finish. As the gears grind, it's generating the design for the data model design for you. There's a summary of what it did, how many statements it imported and so forth. You can save the summary of the import operation, or you can close it. In this case, I'm going to close it. What you see is the imported data model, including the foreign key relationship between the orders and order items table. You will also be able to use SQL Developer long after the data model has been deployed, for report writing, ad hoc queries, and DBA specific duties, such as managing users, table spaces, and database server resources.

Here's a built-in report to list the columns for all views in the database. An Oracle DBA is only as good, and as efficient, as the tool sets they use. Using Oracle's free data modeling, deployment, and development tools, will help any DBA power user be successful in their job. The tools that Oracle provides can truly be one-stop shopping for all of your database design, development, and maintenance needs.