

Slide 1: What Is SQL Anyway?

# What is SQL Anyway?

---

**Sadie St. Lawrence, MSA**  
*Data Scientist, VSP*

**UC DAVIS**  
**EXTENSION**

Slide 2: Learning Objectives

# Learning Objectives

Define SQL

Discuss how SQL differs from other computer languages

Explain how SQL is used in a database

Slide 3: Learning Objectives

## Learning Objectives

Compare and contrast roles of database administrator and data scientist

Explain importance of knowing which SQL syntax you're using in a given database

Slide 4: What Is SQL?

## What is SQL?

Structured Query Language (SQL) is a standard computer language for relational database management and data manipulation

Used to query, insert, update and modify data

Pronounced as “sequel” or S-Q-L

Slide 5: What Is SQL?

## What is SQL?

Used to **communicate** with databases

Statements are made up of **descriptive words** and are easy to learn

SQL is a **non-procedural language**:

- Cannot write complete applications
- Simple, but powerful

Slide 6: How is SQL Used?

## How is SQL Used?

SQL is all about **data**!

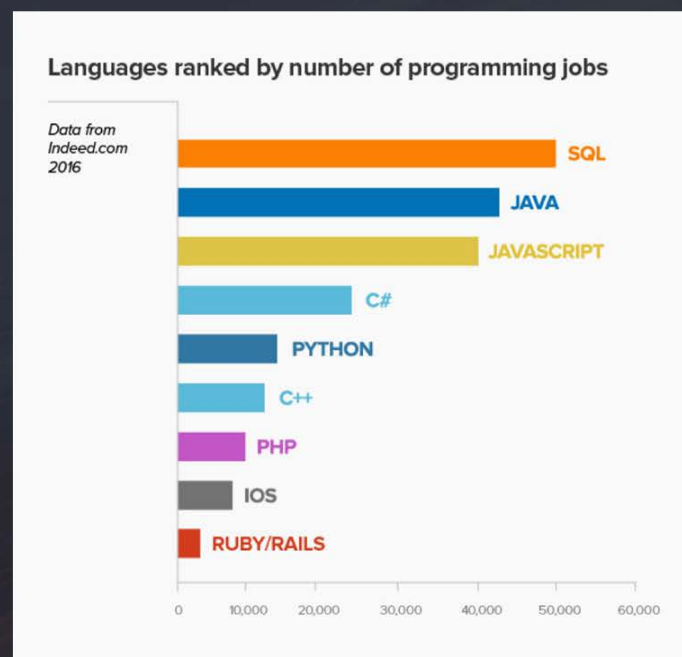
**Read/retrieve data**

**Write data** – add data to a table

**Update data** – insert new data

Slide 7: Who Uses SQL?

# Who Uses SQL?



Slide 8: Who Uses SQL?

## Who Uses SQL?

Backend Developer    Data Architect

QA Engineer            ETL Developer

Database Admin  
(DBA)                    Systems Engineer

Data Scientist

Data Analyst

System Admin



Slide 9: Database Administrator or Data Scientist

## Database Administrator or Data Scientist

### Database Administrator

Manages/governs entire database

Gives permissions to users

Determines access to data

Manages and creates tables

Uses SQL to query and retrieve data

### Data Scientist

End user of a database

Uses SQL to query and retrieve data

Slide 10: How Do Data Scientists Use SQL?

## How Do Data Scientists Use SQL?

Retrieve data

May create their own table or test environment

Combine multiple sources together

Writes complex queries for analysis

Slide 11: SQL and Database Management Systems

## SQL and Database Management Systems

How you write syntax will depend on what DBMS you are using

Each DBMS has its own “**dialect**”

SQL can **translate**

You will tweak based on the “**dialect**” your DBMS speaks

Slide 12: Relational Database Management Systems

## Relational Database Management Systems

SQL Server

Microsoft SQL Server

IBM DB2 Oracle

Apache Open Office  
Base

Sybase ASE

SQLite

PostgreSQL

MySQL

Slide 1: Data Models Part 1: Thinking about Your Data

# Data Models Part 1: Thinking about Your Data

---

**Sadie St. Lawrence, MSA**  
*Data Scientist, VSP*

**UCDAVIS**  
**EXTENSION**

Slide 2: Learning Objectives

## Learning Objectives

Explain why thinking before coding is important

Explain why it is important to understand how the data in a database relates to one another

Describe what a database is

Slide 3: Think Before You Code

# Think Before You Code

What is the **problem** you are  
trying to **solve**?

Slide 4: Understand Your Data

## Understand Your Data

Understand the **business process** or **subject matter** the data is modeled after

Know the **business rules**

Understand how your data is **organized and structured** in the table (modeled)



Slide 5: Why This is Worthwhile

## Why This is Worthwhile

Get more **accurate** results

**Speed** up your work

Have less **rework**

Slide 6: Databases and Tables

# Databases and Tables

**Database:** A container (usually a file or set of files) to store organized data; a set of related information

**Tables:** A structured list of data or a specific type



Slide 7: Columns and Rows

# Columns and Rows

**Column:** A single field in a table - all tables are made up of one or more columns

**Row:** A record in a table

	A	B	C	D	E	F	G	H	I	J	K
1	sku	name	brand	Price	item_height	item_length	item_width	item_unit_o	item_weight	item_unit_of_weight	
2	PFI-F0314BPY	Pfister F-031-4	Pfister	129.99	3.8	20.6	12.7	inches	6.5	pounds	
3	PFI-F042HAKO	Pfister Amher	Pfister	79.99	7.69	4.81	6.56	inches	3.42	pounds	
4	PFI-FWK1340	Pfister F-WK1-	Pfister	119.99	2.5	18.5	10	inches	5.12	pounds	
5	PFI-GT343TCC	Pfister GT34-3	Pfister	149.99	2.5	18.5	10	inches	6.2	pounds	
6	PFI-MP8LNKK	Pfister Langst	Pfister	99.99	7.67	14.22	7.67	inches	3.2	pounds	
7	B00B4QEPOU	Pfister GT529-	Pfister	109.99	2.5	18.5	10	inches	3.17	pounds	
8	PFI-GT529DCC	Pfister GT529-	Pfister	179.99	2.5	24.5	10.5	inches	3.17	pounds	
9	PFI-GT529DSS	Pfister GT529-	Pfister	114.99	2.5	24.5	10.5	inches	3.17	pounds	
10	B007LEP02Q	Pfister F-031-4	Pfister	249.99	4.1	20.6	12.8	inches	6.5	pounds	
11	PFI-F0314BPS	Pfister F-031-4	Pfister	144.99	4.1	20.6	12.8	inches	6.5	pounds	
12											

Slide 1: The Evolution of Data Models

# The Evolution of Data Models

---

**Sadie St. Lawrence, MSA**  
*Data Scientist, VSP*

**UCDAVIS**  
**EXTENSION**

Slide 2: Learning Objectives

# Learning Objectives

Describe data modeling

Define relational database system

Discuss advent of relational  
databases in SQL

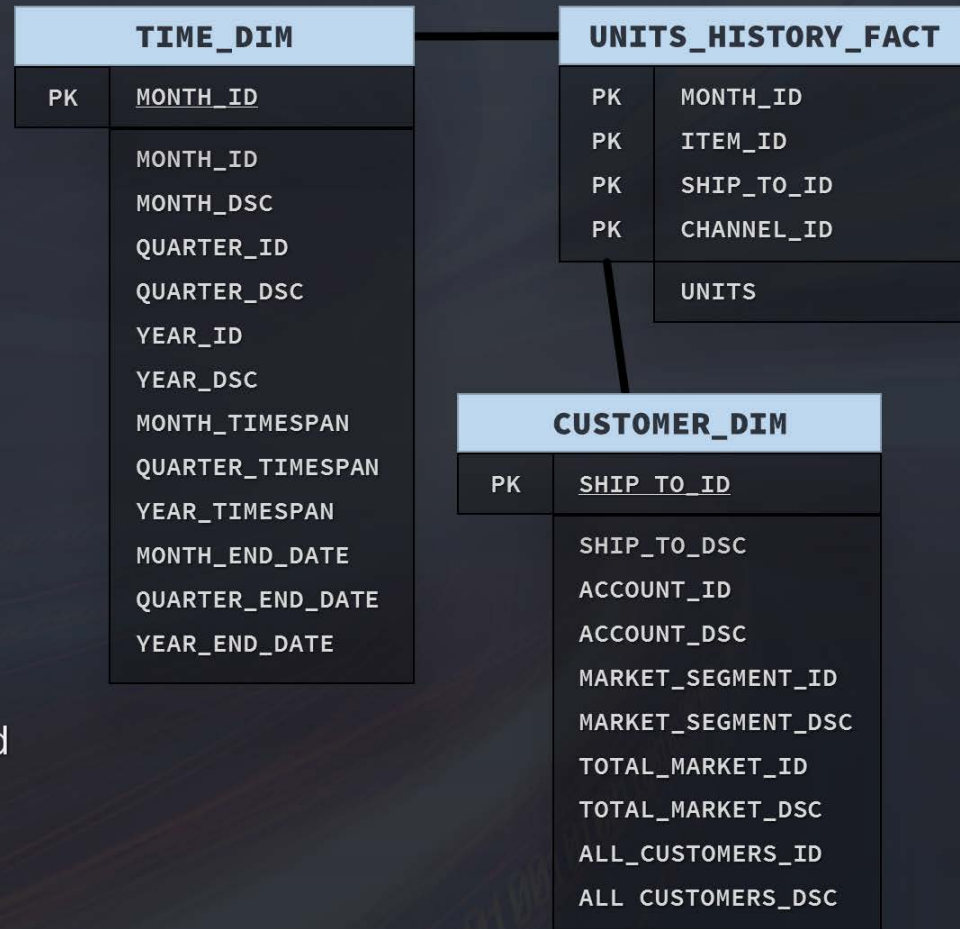
Slide 3: What Is Data Modeling

# What is Data Modeling?

Organizes and structures information into multiple, related tables

Can represent a business process or show relationships between business processes

Should closely represent real world



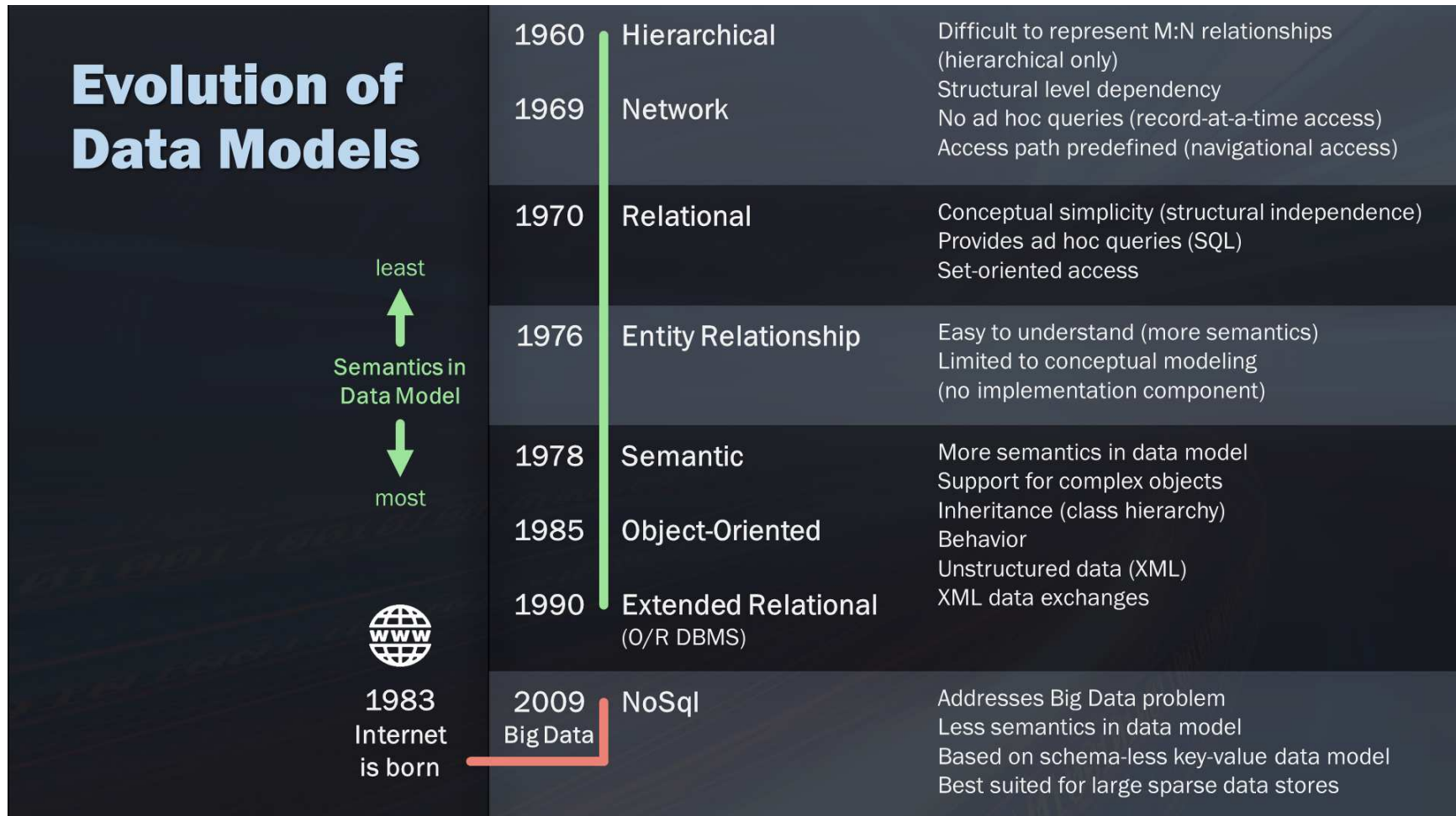
Slide 4: Types of Data Models

## Types of Data Models

Models for **prediction** built by data scientists

Data model as **data tables** represented and organized in a database

Slide 5: Evolution of Data Models





Slide 6: SQL in a Big Data World

## SQL in a Big Data World

NoSQL – Not Only SQL

A mechanism for storage and retrieval of unstructured data modeled by means other than tabular relations in relational databases

Slide 1: Data Models Part 3: Relational vs. Transactional Models

# Data Models Part 3: Relational vs. Transactional Models

---

**Sadie St. Lawrence, MSA**  
*Data Scientist, VSP*

**UC DAVIS**  
**EXTENSION**

Slide 2: Learning Objectives

## Learning Objectives

Define and describe both relational and transactional database models

Define entities, attributes, and relationships

Describe and explain the differences between a one-one, one-many, and many-many relationships

Slide 3: Learning Objectives

## Learning Objectives

Describe primary keys in a database

Explain how ER diagrams are used to document and illustrate relationships

Slide 4: Relational vs. Transactional Model

# Relational vs. Transactional Model

## Relational Model

Allows for easy querying and data manipulation in an easy, logical and intuitive way

## Transactional Model

Operational database – insurance claims within a healthcare database

Slide 5: Data Model Building Blocks

# Data Model Building Blocks

## Entity:

Person, place thing or event  
Distinguishable, unique, and distinct

## Attribute:

A characteristic of an entity

## Relationship:

Describes association  
among entities

- One-to-many
- Many-to-many
- One-to-one

Slide 6: Data Model Building Blocks

## Data Model Building Blocks

**One-to-many:** customer to invoices

**Many-to-many:** student to classes

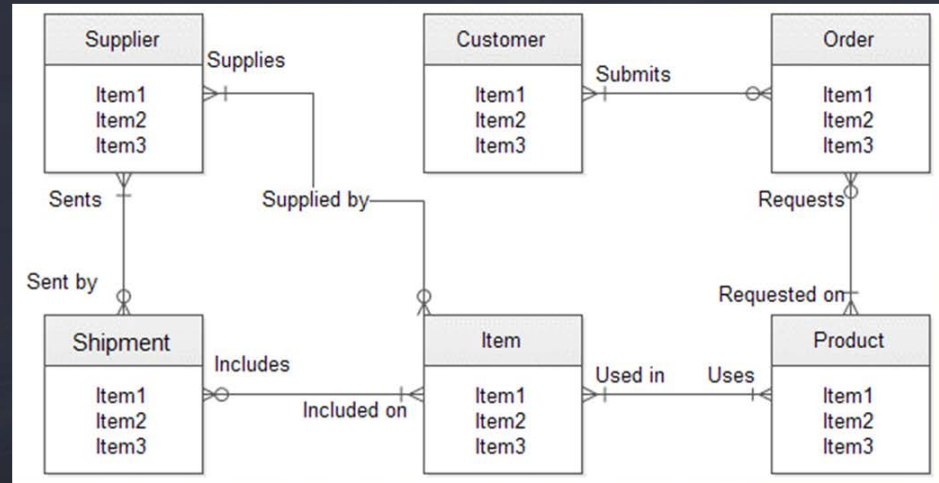
**One-to-one:** manager to store

## Slide 7: ER Diagrams

# ER Diagrams

## ER model

Is composed of entity types and specifies relationships that can exist between instances of those entity types





Slide 8: ER Diagrams

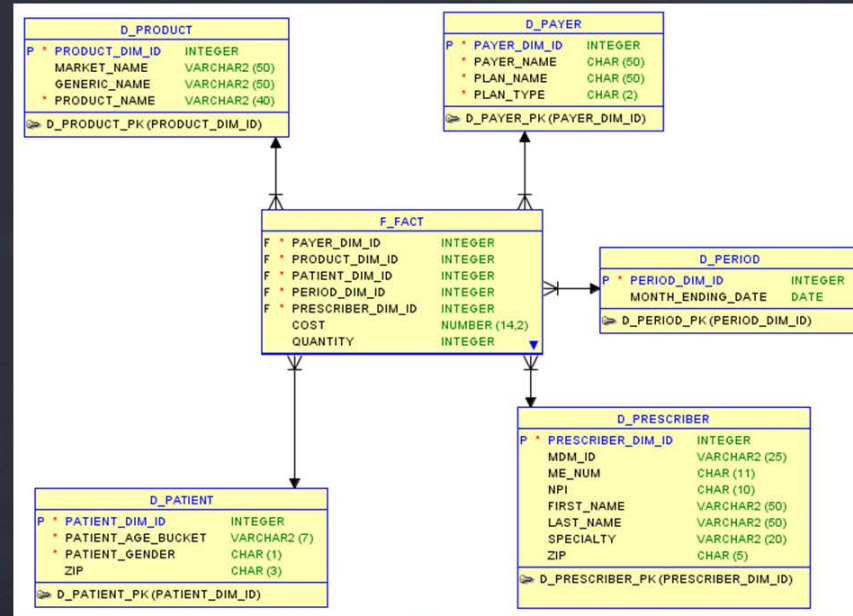
# ER Diagrams

Show relationships

Business process

Represented visually

Show links (primary keys)



## Slide 9: Primary Keys and Foreign Keys

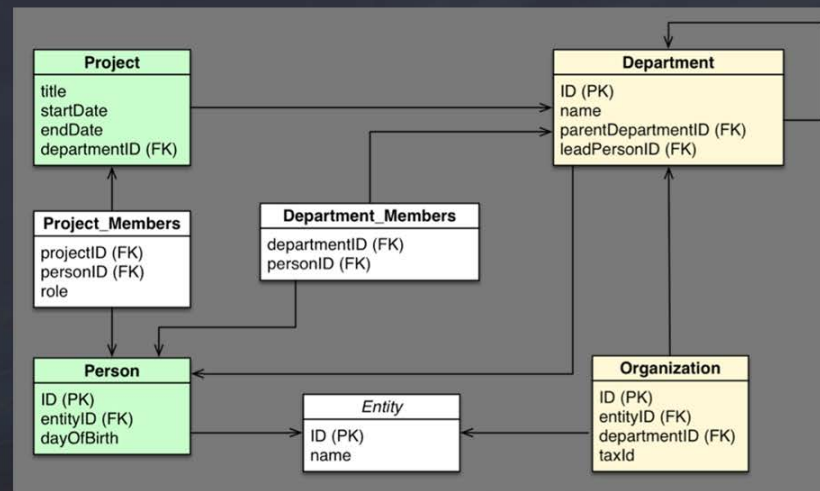
# Primary Keys and Foreign Keys

## Primary Key

A column (or set of columns) whose values uniquely identify every row in a table

## Foreign Key

One or more columns that can be used together to identify a single row in another table



Slide 10: ER Diagram Notation

# ER Diagram Notation

Chen Notation

Crow's Foot Notation

UML Class Diagram Notation

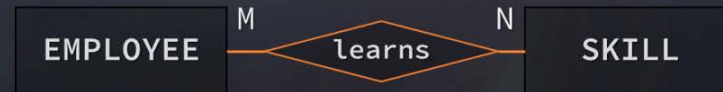
## Slide 11: Chen Notation

# Chen Notation

A **One-to-Many (1:M) Relationship**: a PAINTER can paint many PAINTINGs; each PAINTING is painted by one PAINTER.



A **Many-to-Many (M:N) Relationship**: an EMPLOYEE can learn many SKILLs; each SKILL can be learned by many EMPLOYEEs.



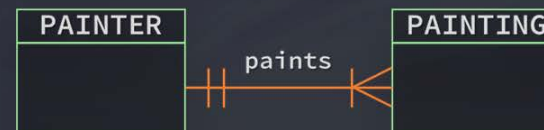
A **One-to-One (1:1) Relationship**: an EMPLOYEE manages one STORE; each STORE is managed by one EMPLOYEE.



## Slide 12: Crow's Good Notation

## Crow's Foot Notation

**A One-to-Many (1:M) Relationship:** a PAINTER can paint many PAINTINGs; each PAINTING is painted by one PAINTER.



**A Many-to-Many (M:N) Relationship:** an EMPLOYEE can learn many SKILLs; each SKILL can be learned by many EMPLOYEEs.



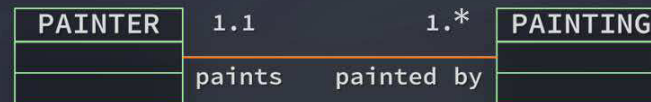
**A One-to-One (1:1) Relationship:** an EMPLOYEE manages one STORE; each STORE is managed by one EMPLOYEE.



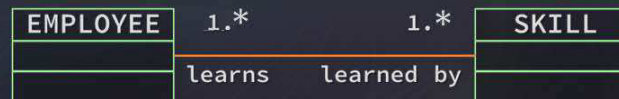
## Slide 13: UML Class Diagram Notation

# UML Class Diagram Notation

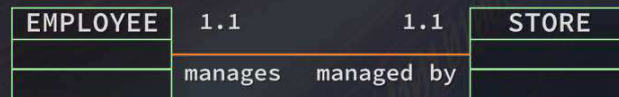
**A One-to-Many (1:M) Relationship:** a PAINTER can paint many PAINTINGs; each PAINTING is painted by one PAINTER.



**A Many-to-Many (M:N) Relationship:** an EMPLOYEE can learn many SKILLs; each SKILL can be learned by many EMPLOYEEs.



**A One-to-One (1:1) Relationship:** an EMPLOYEE manages one STORE; each STORE is managed by one EMPLOYEE.



Slide 14: ER Diagram Notation

# ER Diagram Notation

## Chen Notation

A One-to-Many (1:M) Relationship: a PAINTER can paint many PAINTINGs; each PAINTING is painted by one PAINTER.



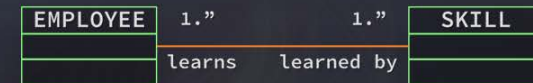
## Crow's Foot Notation



## UML Class Diagram Notation



A Many-to-Many (M:N) Relationship: an EMPLOYEE can learn many SKILLs; each SKILL can be learned by many EMPLOYEEs.



A One-to-One (1:1) Relationship: an EMPLOYEE manages one STORE; each STORE is managed by one EMPLOYEE.



Slide 1: Retrieving Data with a SELECT Statement

# Retrieving Data with a **SELECT** Statement

---

**Sadie St. Lawrence, MSA**  
*Data Scientist, VSP*

**UCDAVIS**  
**EXTENSION**



Slide 2: Learning Objectives

## Learning Objectives

Write a basic **SELECT** statement

Tell a database which table your data will come **FROM**

**SELECT** either all or particular columns from a table in a query

Limit the amount of data which is returned in a query

## Slide 3: The SELECT Statement

# The SELECT Statement

Need to specify two pieces of information to use a SELECT statement: what you want and where you want to select it from.

```
SELECT prod_name  
FROM Products;
```

## Output

```
prod_name  
Shampoo  
Toothpaste  
Deodorant  
Toothbrush
```

## Slide 4: Retrieving Multiple Columns

## Retrieving Multiple Columns

Add multiple column names, be sure to use a comma

```
SELECT prod_name, prod_id,  
       prod_price
```

```
FROM Products;
```

```
SELECT   prod_name  
        ,prod_id  
        ,prod_price
```

```
FROM Products;
```

## Slide 5: Retrieving Multiple Columns Using a Wildcard

## Retrieving Multiple Columns Using a Wildcard

Request all columns by using the asterisk (\*) wildcard character instead of column names

```
SELECT *  
FROM Products;
```

Slide 6: Why Limit Results?

## Why Limit Results?

If your database is large

You might only want to see a sample of the data

```
SELECT columns you wish to see  
FROM specific table  
LIMIT number of records
```

## Slide 7: Limiting Results Using Different Syntaxes

# Limiting Results Using Different Syntaxes

## SQLite

```
SELECT prod_name  
FROM Products  
LIMIT 5;
```

## Oracle

```
SELECT prod_name  
FROM Products  
WHERE ROWNUM <=5;
```

## DB2

```
SELECT prod_name  
FROM Products  
FETCH FIRST 5 ROWS ONLY;
```

Slide 1: Creating Tables

# Creating Tables

---

**Sadie St. Lawrence, MSA**  
*Data Scientist, VSP*

**UC DAVIS**  
**EXTENSION**

Slide 2: Learning Objectives

## Learning Objectives

Discuss situations where it's beneficial to create new tables

Create new tables within an existing database

Write data to a new table

Defining whether columns can accept **NULL** values or not.



Slide 3: Why Tables are Useful

## Why Tables are Useful

Use tables to make models and predictions

Create dashboards

Visualize data with other tools

Extract data from other sources

## Slide 4: Creating Your Own Table

## Creating Your Own Table

```
CREATE TABLE Shoes
```

```
(  
  Id          char(10)      PRIMARY KEY,  
  Brand       char(10)      NOT NULL,  
  Type        char(250)     NOT NULL,  
  Color       char(250)     NOT NULL,  
  Price       decimal(8,2)  NOT NULL,  
  Desc        Varchar (750) NULL  
);
```

## Slide 5: Nulls and Primary Keys

## Nulls and Primary Keys

Every column is either NULL  
or NOT NULL

An error will be returned if one tries  
to submit a column with no value

Don't confuse null values with  
empty strings

Primary keys can not be null

```
CREATE TABLE Shoes
(
  Id          char(10)          PRIMARY KEY,
  Brand       char(10)          NOT NULL,
  Type        char(250)         NOT NULL,
  Color       char(250)         NOT NULL,
  Price       decimal(8,2)      NOT NULL,
  Desc        Varchar (750)     NULL
);
```

Slide 6: Primary Keys and Columns Cannot Be NULL

## Primary Keys and Columns Cannot Be NULL

Primary Keys **MUST** have a value

```
CREATE TABLE Shoes
(
  Id          char(10)          PRIMARY KEY,
  Brand       char(10)          NOT NULL,
  Type        char(250)         NOT NULL,
  Color       char(250)         NOT NULL,
  Price       decimal(8,2)      NOT NULL,
  Desc        Varchar (750)     NULL
);
```

## Slide 7: Adding Data to the Table

## Adding Data to the Table

```
INSERT INTO Shoes
VALUES ('14535974'
      , 'Gucci'
      , 'Slippers'
      , 'Pink'
      , '695.00'
      , NULL
      );
```

## Slide 8: Adding Data to the Table

## Adding Data to the Table

```
INSERT INTO Shoes
VALUES ('14535974'
      , 'Gucci'
      , 'Slippers'
      , 'Pink'
      , '695.00'
      , NULL
      );
```

```
INSERT INTO Shoes
      (Id
      ,Brand
      ,Type
      ,Color
      ,Price
      ,Desc
      )
VALUES
      ('14535974'
      , 'Gucci'
      , 'Slippers'
      , 'Pink'
      , '695.00'
      , NULL
      );
```

Slide 1: Creating Temporary Tables

# Creating Temporary Tables

---

**Sadie St. Lawrence, MSA**  
*Data Scientist, VSP*

**UCDAVIS**  
**EXTENSION**

Slide 2: Learning Objectives

## Learning Objectives

Create temporary tables

Describe limitations of temporary tables

Discuss strategies for researching syntax for particular database management systems



Slide 3: Why Create Temporary Tables?

## Why Create Temporary Tables?

Temporary tables will be deleted when current session is terminated

Faster than creating a real table

Useful for complex queries using subsets and joins

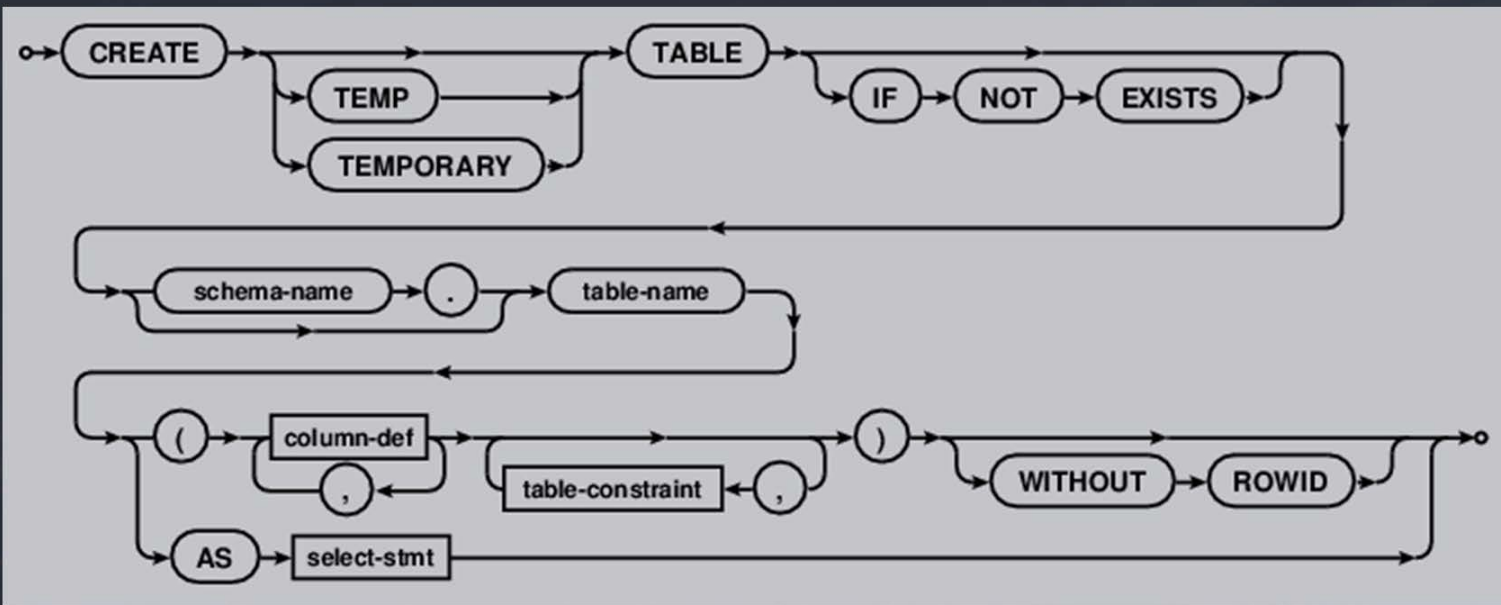
## Slide 4: How to Create a Temporary Table

## How to Create a Temporary Table

```
CREATE TEMPORARY TABLE Sandals AS  
(  
    SELECT *  
    FROM shoes  
    WHERE shoe_type = 'sandals'  
)
```

Slide 5: Creating a Table in SQL Lite

# Creating a Table in SQLite



Slide 6: Primary Keys and Columns Cannot Be NULL

# Never Stop Learning



Key words: Statement, RDBMS

Research how to:

Update tables

Delete tables

Slide 1: Adding Comments to SQL

# Adding Comments to SQL

---

**Sadie St. Lawrence, MSA**  
*Data Scientist, VSP*

**UCDAVIS**  
**EXTENSION**

Slide 2: Learning Objectives

## Learning Objectives

Discuss importance of writing comments as a part of your code

Describe several comment syntaxes used in SQL

Write comments in your code

Slide 3: Why Add Comments

# Why Add Comments

Help you remember **what** you were doing  
and **why**

Mute the expression of code  
(commenting out code)

Troubleshoot query issues systematically

## Slide 4: Adding Comments

# Adding Comments

## Single Line

```
SELECT shoe_id  
- -,brand_id  
,shoe_name  
from shoes
```

## Section

```
SELECT shoe_id  
/*,brand_id  
,shoe_name  
*/  
from shoes
```



## Slide 5: Comments Gone Wrong

## Comments Gone Wrong

```
SELECT
--Getting the average units for each material
  material
  ,avg(netunits) AS avg_netunits
--Getting this from the sku table for shoes
FROM shoe_skoe
--Grouping it by each month of the year first, and then
material
  GROUP BY yearmonth, material
--Ordering it in Descending order.
  ORDER BY avg_netunits DESC;
```

Slide 6

```
--Selecting the average amount for shirts
SELECT avg(c.Shirt_amount)
--Joining to the customer table and the dates table so we can provide
filters.
FROM Shirt c INNER JOIN Customer m ON c.shirt_nbr = m.shirt_nbr
        INNER JOIN dates dt ON c.date_skey = dt.date_skey
--Filtering on dates for 2014 and filtering
WHERE date_yyyymm between '201401' AND '201412'
---Need to create the customers age from their date of birth
        AND YEAR(CURRENT DATE - Customer_DOB) >= 0
        AND YEAR(CURRENT DATE - Customer_DOB) < 19
        AND c.sbtck = 1
        AND c.category in ('A','9')
--Ordering it by random
ORDER BY rand()
        FETCH first 10000 rows only;
```

Slide 7: Source Code Editors

## Source Code Editors

Environment **separate from the database**  
where you can write and edit code

e.g.: Notepad++

Automatically highlights and indents  
statements

Helps you write clean code