Syracuse University
School of Information Studies

# Introduction to SQL

# Agenda

- What is SQL?
- History of SQL
- Naming conventions
- SQL data definition
- SQL data manipulation
- Schema inspection

Syracuse University
School of Information Studies

Introduction to SQL

# The End

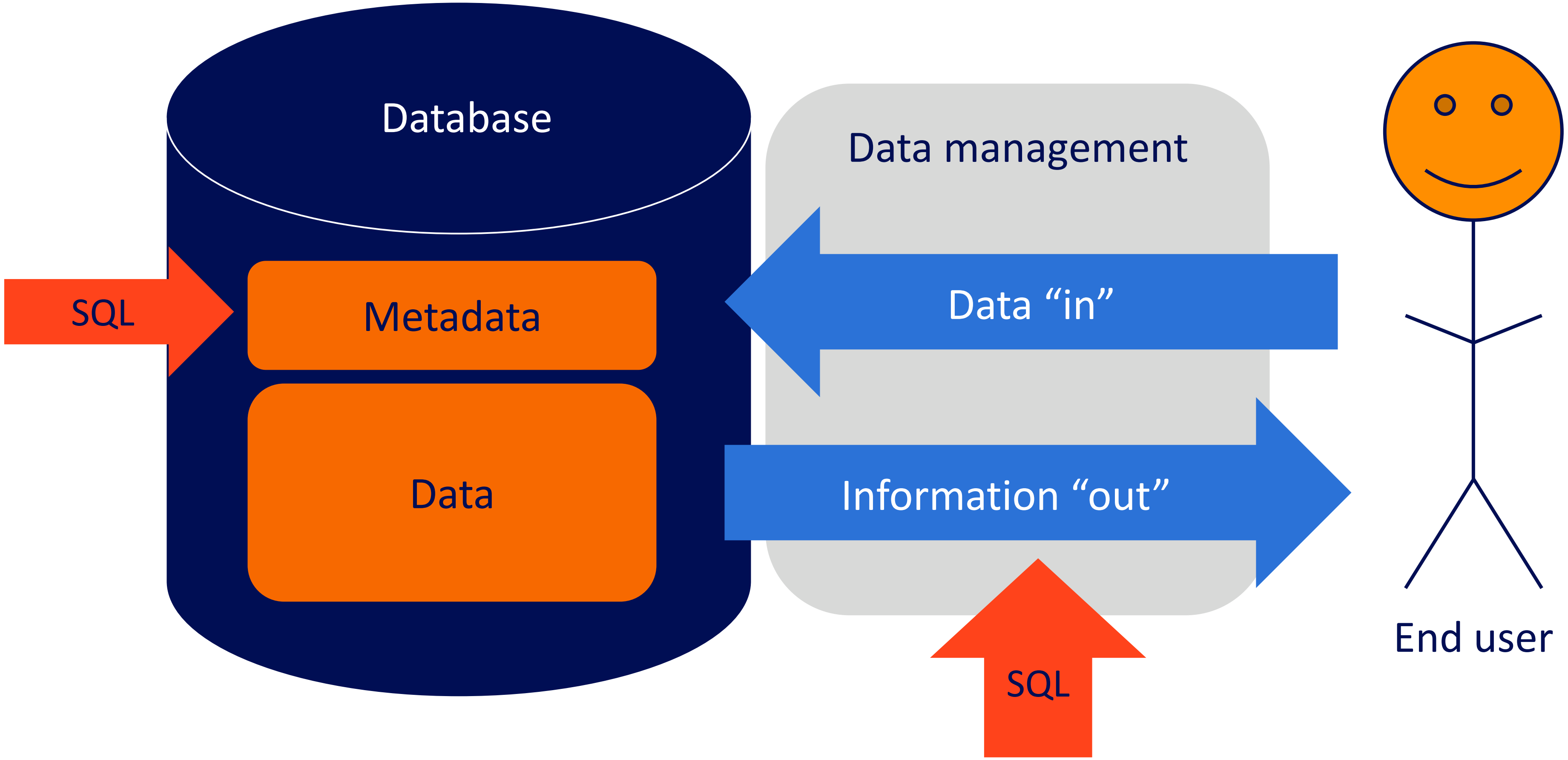
Syracuse University
School of Information Studies

# What Is SQL?

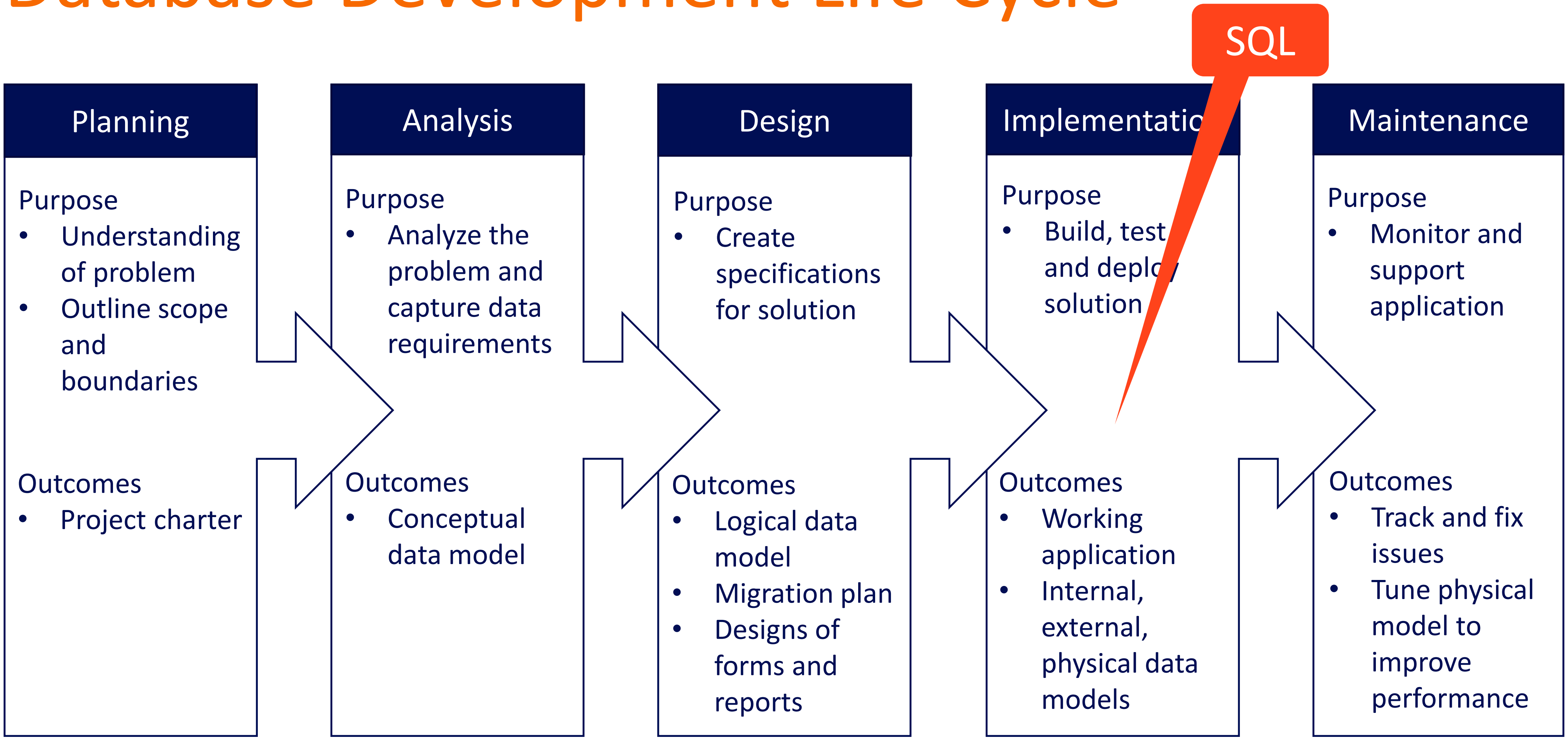
# What Is SQL?

SQL, or Structured Query Language, is a domain-specific programming language for database management systems.

# SQL Is Domain Specific to Databases



# Database Development Life Cycle



SQL

## Planning

Purpose

- Understanding of problem
- Outline scope and boundaries

Outcomes

- Project charter

## Analysis

Purpose

- Analyze the problem and capture data requirements

Outcomes

- Conceptual data model

## Design

Purpose

- Create specifications for solution

Outcomes

- Logical data model
- Migration plan
- Designs of forms and reports

## Implementatio

Purpose

- Build, test and deploy solution

Outcomes

- Working application
- Internal, external, physical data models

## Maintenance

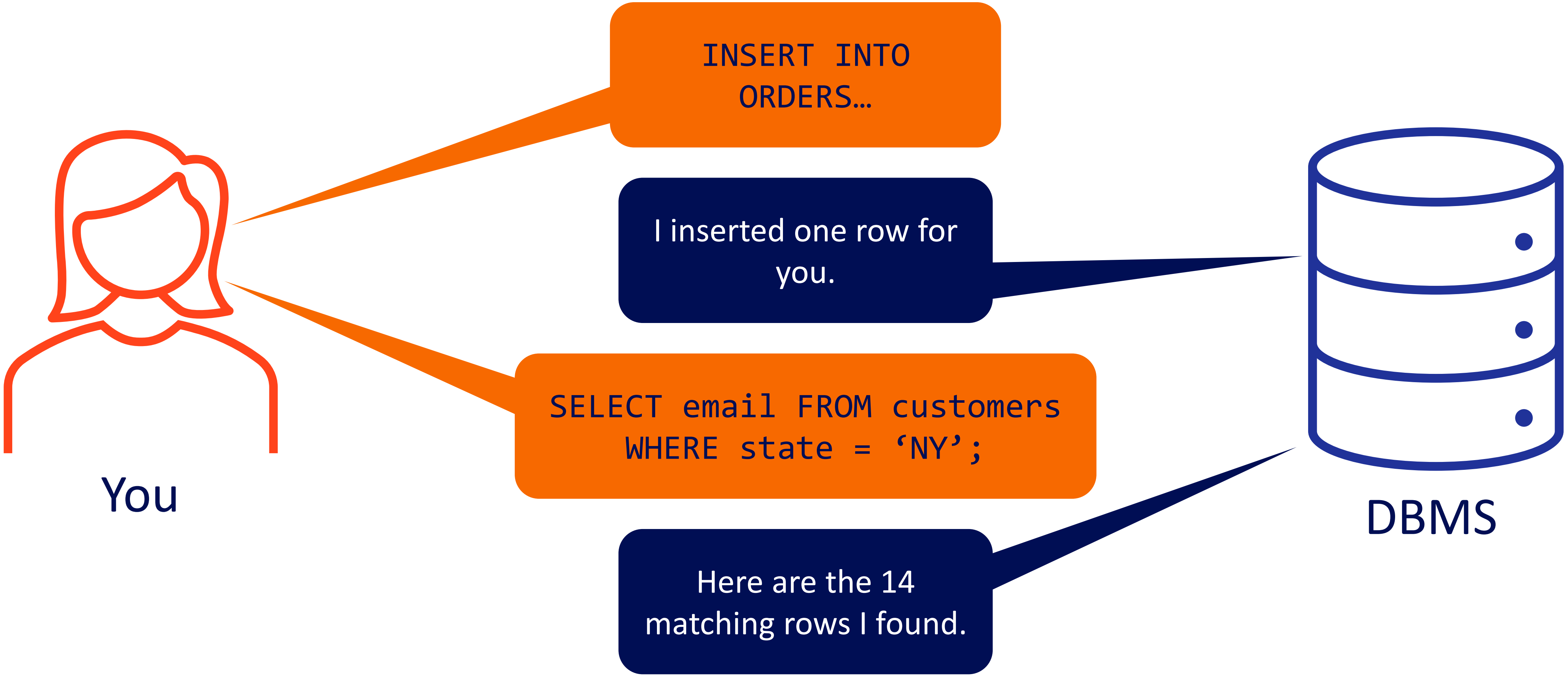
Purpose

- Monitor and support application

Outcomes

- Track and fix issues
- Tune physical model to improve performance

# The Declarative Nature of SQL



You

```
INSERT INTO
ORDERS…
```

I inserted one row for you.

```
SELECT email FROM customers
WHERE state = ‘NY’;
```

Here are the 14 matching rows I found.

DBMS

# SQL History and Standard

- SQL was developed by Chamberlin and Boyce at IBM in 1970.
- It was a query language based on Codd's relational design paper.
- Standardized in 1986, there have been several revisions.
- Each revision adds new features to the standard.
- Not all DBMS vendors follow the standard.

https://en.wikipedia.org/wiki/SQL

Syracuse University
School of Information Studies

## What Is SQL?

# The End

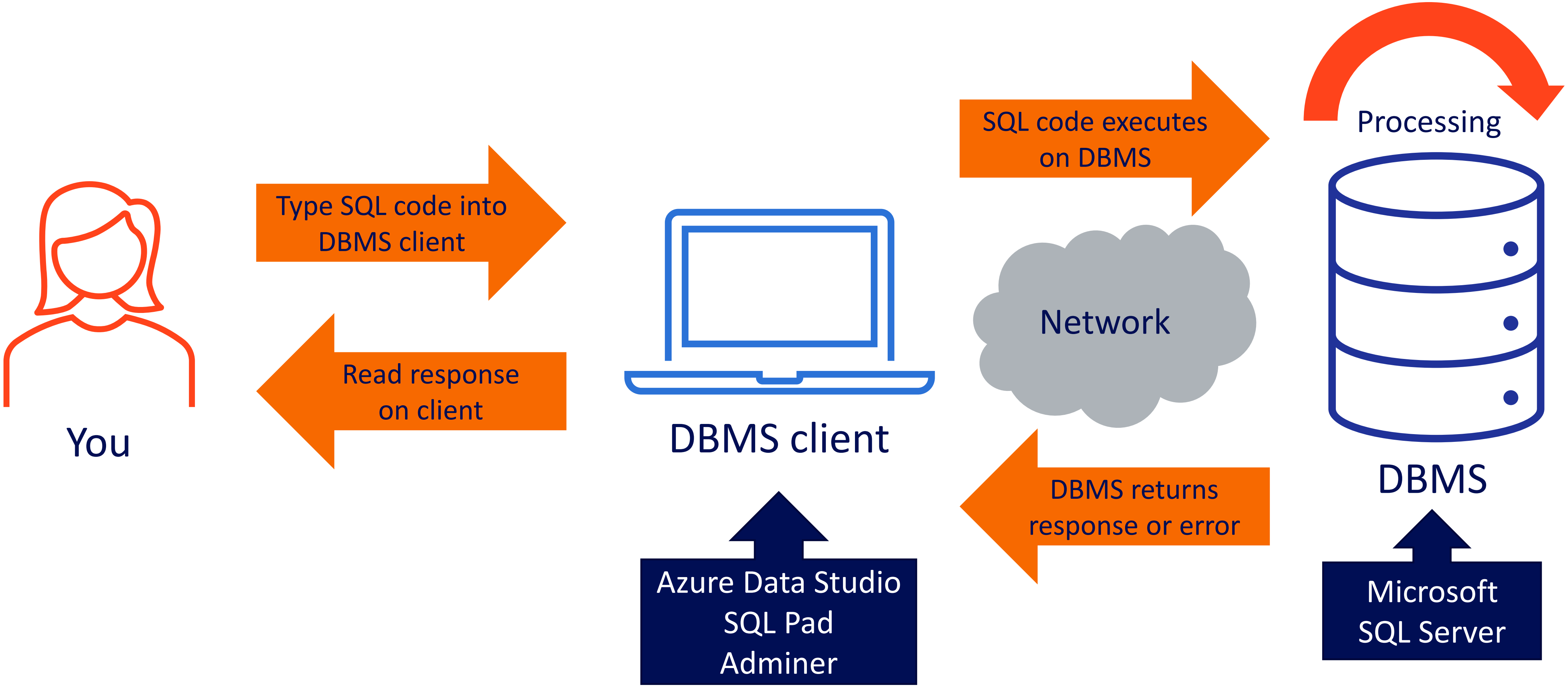
Syracuse University
School of Information Studies

# DBMS and Databases

# Databases in the DBMS

- The database management system manages one or more databases.
- The running DBMS is referred to as an instance.
- We connect to the DBMS instance via a client. The client is used to write the commands execute command.
- The databases are independent collections of data and metadata.

# Your Interactions With the DBMS



Syracuse University
School of Information Studies

DBMS and Databases

# The End

Syracuse University
School of Information Studies

# Demo

Azure Data Studio

# Demo Azure Data Studio

- Let’s connect to our Microsoft SQL Server instance using the Azure Data Studio client
- Let’s view the different databases on our instance
- Using the use command to switch databases programmatically
- Using the go command to batch commands in a single file

Syracuse University
School of Information Studies

Demo: Azure Data Studio

# The End

Syracuse University
School of Information Studies

# SQL Language and Naming Conventions

# SQL Language Categories DDL and DML

- DDL: data definition language
- For object (metadata) management
- Commands: CREATE, ALTER, DROP

- DML: data manipulation language
- For data management, CRUD
- Commands: INSERT, SELECT, UPDATE, SELECT

# Naming Conventions

- Naming conventions are generally agreed-upon schemes for naming objects in a programing language.
- They provide consistency and/or include descriptive metadata.
  - Consistency: How do we name grad students?
    - `GRAD_STUDENTS, grad_students, GradStudents, gradStudents`
  - Metadata: indication of object type
    - `pk_grad_students vs grad_students`
- Adopting a naming convention is paramount.

# Our SQL Object Naming Conventions

| What | Our convention | Rationale for use of this convention |
|---|---|---|
| **All object names** | Use lower case letters only | Disambiguation of database objects<br>Example: EMPLOYEE vs. employee |
| **All object names** | Use underscore in place of **space** | Avoids the need to place brackets around identifiers<br>Example: customer_email vs. [customer email] |
| **Tables** | Pluralize | Since a table contains many “things” it should be plural<br>Example: employees vs. employee |
| **Column names** | Qualify with table name | When learning SQL, helps you define table scope easily<br>Example: vendor_zipcode vs. employee_zipcode |
| **Constraints** | pk=primary key, fk=foreign key u=unique, ck=check, i=index | Disambiguation of constraint names<br>Example: ck_vendor_zipcode vs. fk_vendor_zipcode |
| **Constraint dependencies** | Dependent object included | Helps with disambiguation of constraint to table<br>Example: u_customer_email vs. u_vendor_email |

Syracuse University
School of Information Studies

# SQL Language and Naming Conventions

## The End

Syracuse University
School of Information Studies

# SQL DDL

# SQL DDL Commands

- The SQL data definition language commands allow us to manage metadata objects in our database.
  - CREATE creates an object, such as a table.
  - ALTER changes an existing object.
  - DROP deletes an existing object.
- The DMBS has many kinds of objects. For starters, the only objects we will manage are tables.

Syracuse University
School of Information Studies

SQL DDL

# The End

Syracuse University
School of Information Studies

# SQL CREATE TABLE

# SQL: CREATE TABLE

```
CREATE TABLE your_table_name (
  table_column_name datatype NULL | NOT NULL,
  next_table_column_name datatype NULL | NOT NULL,
  (repeat for as many columns as you have),
  CONSTRAINT pk_table_column_name
    PRIMARY KEY (column_name_of_pk)
)
```

- Items in UPPER CASE are keywords that are part of the SQL language.
- Items in italic should be replaced by actual object names.
- The pipe | means choose between NULL or NOT NULL.

# SQL Data Types

| Data type | Description |
|---|---|
| Exact numeric types | |
| Int | 4-byte integers in the range -2^31 to +2^31-1 |
| smallint | 2-byte integers in the range -32767 to +32768 |
| Tiny | 1-byte integers in the range 0 to 255 |
| Bigint | 8-byte integers in the range -2^63 to +2^63-1 |
| decimal(p,s) | Binary coded decimal; p=precision total number of digits, and s=scale to the right of the decimal point |
| numeric(p,s) | Same as decimal (p,s) |
| Bit | The value of 0, 1 or NULL |
| Approximate numeric types | |
| Real | 4-byte floating point number |
| Double | 8-byte floating point number |
| String types | |
| varchar(n) | Varying characters; length no more than n bytes |
| char(n) | Characters of exact length n bytes |
| nvarchar(n) | Varying 2-byte characters; length no more than n bytes; Unicode support |
| nchar(n) | Fixed 2-byte characters of exact length n bytes; Unicode support |
| Date/time types | |
| Date | 3 bytes storage for dates only in range 0001-01-01 to 9999-12-31 |
| Time | 5 bytes storage for time of day including fractional seconds |
| datetime | Combined date and time type, 8 bytes total |

Syracuse University
School of Information Studies

SQL CREATE TABLE

# The End

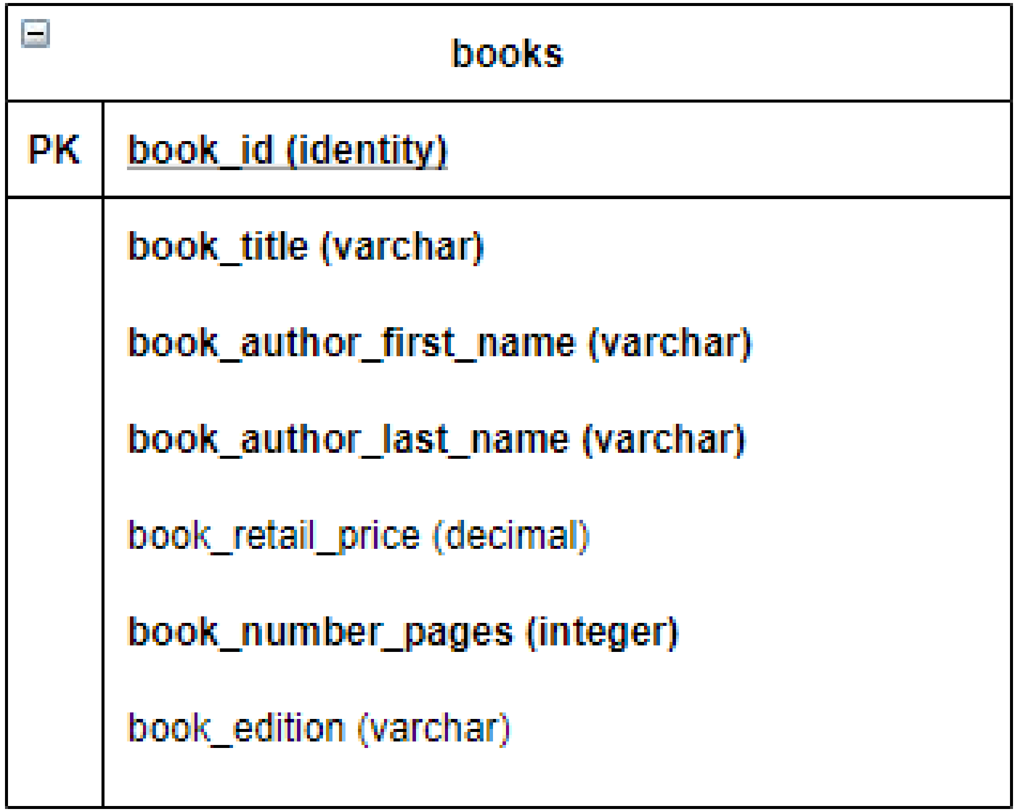
Syracuse University
School of Information Studies

# Demo

CREATE TABLE

# Demo: Creating a Table in SQL

- We will use the Azure Data Studio application
- Let’s create a books table in the demos database
- Logical model →→
- Verify that the table was created using the tool



| | books |
|---|---|
| PK | book_id (identity) |
| | book_title (varchar) |
| | book_author_first_name (varchar) |
| | book_author_last_name (varchar) |
| | book_retail_price (decimal) |
| | book_number_pages (integer) |
| | book_edition (varchar) |

Syracuse University
School of Information Studies

Demo: CREATE TABLE

# The End

Syracuse University
School of Information Studies

# SQL INSERT Statement

# SQL: INSERT

```
INSERT INTO your_table_name
   (table_col1, table_col2, etc.…)
VALUES
   (value1, value2, etc.…)
```

- The values must correspond to the columns.
- Any columns omitted will insert their default value or NULL.
- You can insert additional items by more values.

Syracuse University
School of Information Studies

## SQL INSERT Statement

# The End

Syracuse University
School of Information Studies

# Demo

Inserting Data Into a Table

# Demo: Inserting Data Into a Table

- Let’s insert these data into the books table in the demos database
- Data →→→→

| Title | Author | Price | Pages |
|---|---|---|---|
| The Art of War | Sun Tzu | 9.95 | 260 |
| Frankenstein | Mary Shelly | 14.95 | 280 |
| A Christmas Carol | Chuck Dickens | | 110 |
| The Time Machine | H.G. Wells | 9.95 | 84 |

- Verify that the data were inserted using the tool

Syracuse University
School of Information Studies

# Demo:
# Inserting Data Into a Table

## The End

Syracuse University
School of Information Studies

# SQL ALTER TABLE

# SQL ALTER TABLE

The `ALTER TABLE` statement allows us to change the schema of our table; it comes in three flavors.

1. `ALTER TABLE ADD`: add a column or constraint
2. `ALTER TABLE ALTER`: replace a column or constraint
3. `ALTER TABLE DROP`: delete column or constraint

# SQL: ALTER TABLE… COLUMNS

```
ALTER TABLE table_name ADD COLUMN
column_name datatype NULL|NOT NULL;
ALTER TABLE table_name ALTER COLUMN
column_name datatype NULL|NOT NULL;
ALTER TABLE table_name DROP COLUMN
column_name;
```

# Recall: Data Integrity Constraints

- Primary key constraint: establishes entity integrity within the table; data are ordered by the values in the key
- Unique constraint: functions like a primary key constraint but does not affect the physical order of the data in the table
- Check constraint: an expression that must be true prior to data being written to the database
- Default value constraint: a value used for a data attribute when one is not specified
- Foreign key constraint: establishes referential integrity over the column, ensuring its values are in the set of primary keys from the referring table
- Lookup table: a foreign key constraint with a table to restrict a column to a list of values

# SQL: ALTER TABLE… CONSTRAINT

```
ALTER TABLE table_name ADD CONSTRAINT
constraint_name PRIMARY KEY (column);
ALTER TABLE table_name ADD CONSTRAINT
constraint_name UNIQUE (column);
ALTER TABLE table_name ADD CONSTRAINT
constraint_name CHECK (expression);
```

# SQL: ALTER TABLE ADD CONSTRAINT

```
ALTER TABLE table_name ADD CONSTRAINT
constraint_name DEFAULT (expression) FOR
column;
```

```
ALTER TABLE table_name ADD CONSTRAINT
constraint_name FOREIGN KEY (column)
REFERENCES pk_table_name(pk_column);
```

```
ALTER TABLE table_name DROP CONSTRAINT
constraint_name;
```

# SQL: DROP TABLE

```
DROP TABLE table_name;
```

Deletes the table and the data within it

Syracuse University
School of Information Studies

SQL ALTER TABLE

# The End

Syracuse University
School of Information Studies

# Demo

ALTER TABLE

# Demo: ALTER TABLE

- Let’s create an editions_lookup lookup table in the demo database and insert some editions
- Alter the books table to add some logical domain
  - The natural key book_isbn column
  - Unique constraint on the book_isbn column
  - A default constraint on book_editions of “1st”
  - A check constraint of book_retail_price >=0
  - A check constraint of number_of_pages >0
  - A foreign key constraint on book_editions using the lookup table
- Testing the constraints, with inserts

Syracuse University
School of Information Studies

Demo: ALTER TABLE

# The End

Syracuse University
School of Information Studies

# SQL DML

# SQL Data Manipulation Language

DML

# SQL DML the CRUD Operations

There is an SQL data manipulation command that corresponds to each of the CRUD operations.

- Create: `INSERT`
- Read: `SELECT`
- Update: `UPDATE`
- Delete: `DELETE`

# SQL: SELECT

```
SELECT table_column1,
table_column2, etc.…
   FROM your_table_name
   [WHERE boolean_expression]
   [ORDER BY column]
```

- Items in square brackets [ ] are optional.
- WHERE allows you to filter based on a Boolean (true/false) expression.
- ORDER BY allows you to sort the output.

# SQL: UPDATE

```
UPDATE your_table_name
  SET table_column1 = new_value1
      table_column2 = new_value2,
      etc.…
  [WHERE boolean_expression]
```

- Items in square brackets [ ] are optional.
- WHERE allows you to filter based on a true/false expression.
- ORDER BY allows you to sort the output.

# SQL: DELETE

```
DELETE FROM your_table_name
  [WHERE boolean_expression]
```

If the WHERE clause is omitted, all matching rows are deleted.

# Understanding the Atomicity of an RDBMS

- Any SQL statement that changes data must succeed or fail as a whole, meaning that there are no partial updates.
- For example, in a table of 10,000 rows, if you update all data in single column and one attribute fails a data integrity constraint, none of the updates will succeed.
- Commands operate on sets of data. The command must be applicable to all data within the set.
  - No intermediate state! This is by design!

Syracuse University
School of Information Studies

SQL DML

# The End

Syracuse University
School of Information Studies

# Demo

DML

# Demo: DML

- We will use the ADS application.
- Let’s use the tables in the demo database to perform SELECT, UPDATE, and DELETE commands.
- SELECT books with more than 150 pages.
- SELECT books with no price.
- UPDATE books fixing Dickens.
- UPDATE the price of all books with more than 150 pages.
- INSERT a book then DELETE it.
- Demonstrate atomicity by lowering the price of all books.

Syracuse University
School of Information Studies

Demo: DML

# The End

Syracuse University
School of Information Studies

# INFORMATION_ SCHEMA

# INFORMATION_SCHEMA

- Provides access to DBMS objects (metadata) via the SELECT statement
- Part of the SQL standard since SQL-92
- These are virtual tables called VIEWS
- INFORMATION_SCHEMA.TABLES
- INFORAMTION_SCHEMA.COLUMNS
- INFORMATION_SCHEMA.TABLE_CONSTRAINTS

Syracuse University
School of Information Studies

INFORMATION_SCHEMA

# The End

Syracuse University
School of Information Studies

# Demo

INFORMATION_SCHEMA

# Demo: INFORMATION_SCHEMA

- We will use the ADS application
- Let’s use the tables in the demo database to perform schema inspection
- Tables
- Columns
- Constraints

Syracuse University
School of Information Studies

## Demo: INFORMATION_SCHEMA

# The End

Syracuse University
School of Information Studies

# Up/Down Scripts

# Putting It All Together

- A programmatic approach to managing change in any stateful system like a database
- Up/down scripts
  - Single script to do a persistent action
  - Complementary script to undo said action

# Up or Down?

- Your Up script should change the state of the data and metadata within the database from A to B
- Example up

  -
    ```
    ALTER TABLE a
      ADD c INT NULL;
    UPDATE a SET c = 42;
    ALTER TABLE a
      ALTER COLUMN c INT NOT NULL
    ```

- Your down script should change the state of the data and metadata within the database from B to A
- Example down (undo)

  -
    ```
    ALTER TABLE a
      DROP COLUMN c;
    ```

Syracuse University
School of Information Studies

Up/Down Scripts

# The End

Syracuse University
School of Information Studies

# Demo

Up/Down Scripts

# Demo: Up/Down Scripts

- We will use the ADS web application.
- Let’s use the demo database to demonstrate up/down principles.
- Example: Add the publisher and website to books.

Demo: Up/Down Scripts

# The End

# Summary

# Summary

- SQL is a domain-specific language used by relational DBMS.
- SQL supports metadata management through the data definition language and CREATE, ALTER, and DROP commands.
- SQL supports data management through the data manipulation language and INSERT, SELECT, UPDATE, and DELETE commands.
- Naming conventions are important to tracking the metadata objects created in SQL.
- Schema inspection can be done in SQL with INFORMATION_SCHEMA.

Syracuse University
School of Information Studies

Summary

# The End