Lab4 is up!

-2

Q4Me

-1

Book vs. Slides	CH06 (1st & 2nd Ed.)
Lab	?
Last Weeks	?

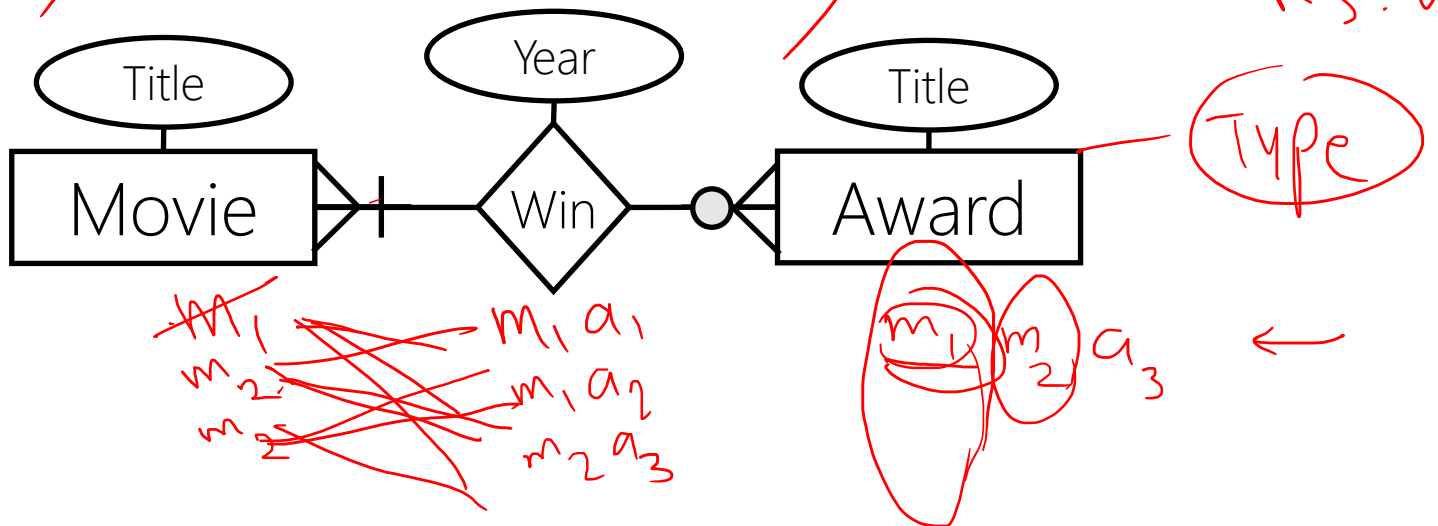
SQL

Q4U

0

Given the following E/R, write relational expression to answer the following questions:

1- Which movie has won an Oscar award? [Movie Title, Award Title, Year]



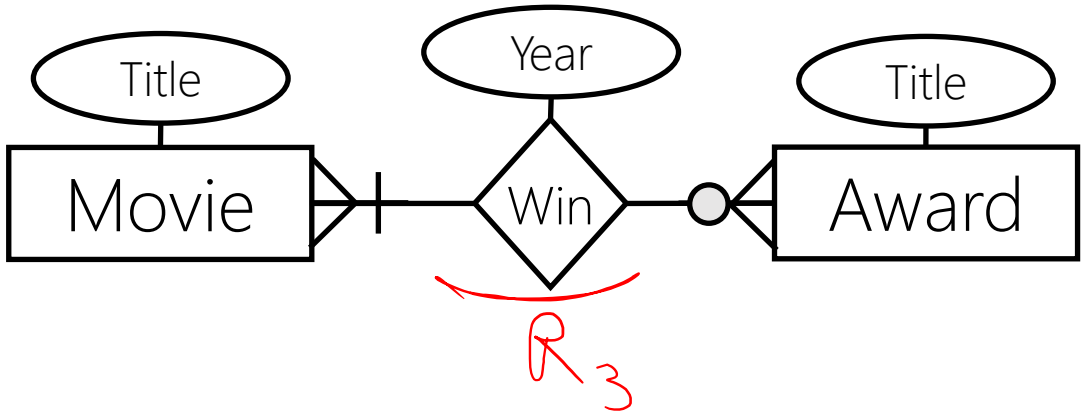
Q4U

0

Given the following E/R, write relational expression to answer the following questions:

2- Which movie has won nothing? [Movie Title]

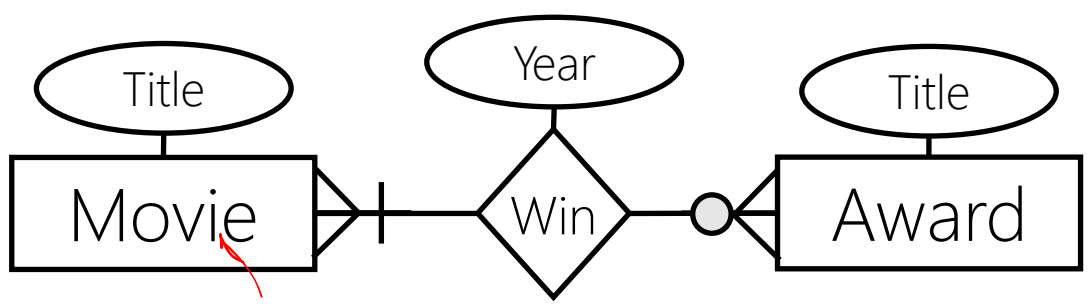
$= \Pi_{Id,Title}(R_1) - \Pi_{MovId}(R_3)$

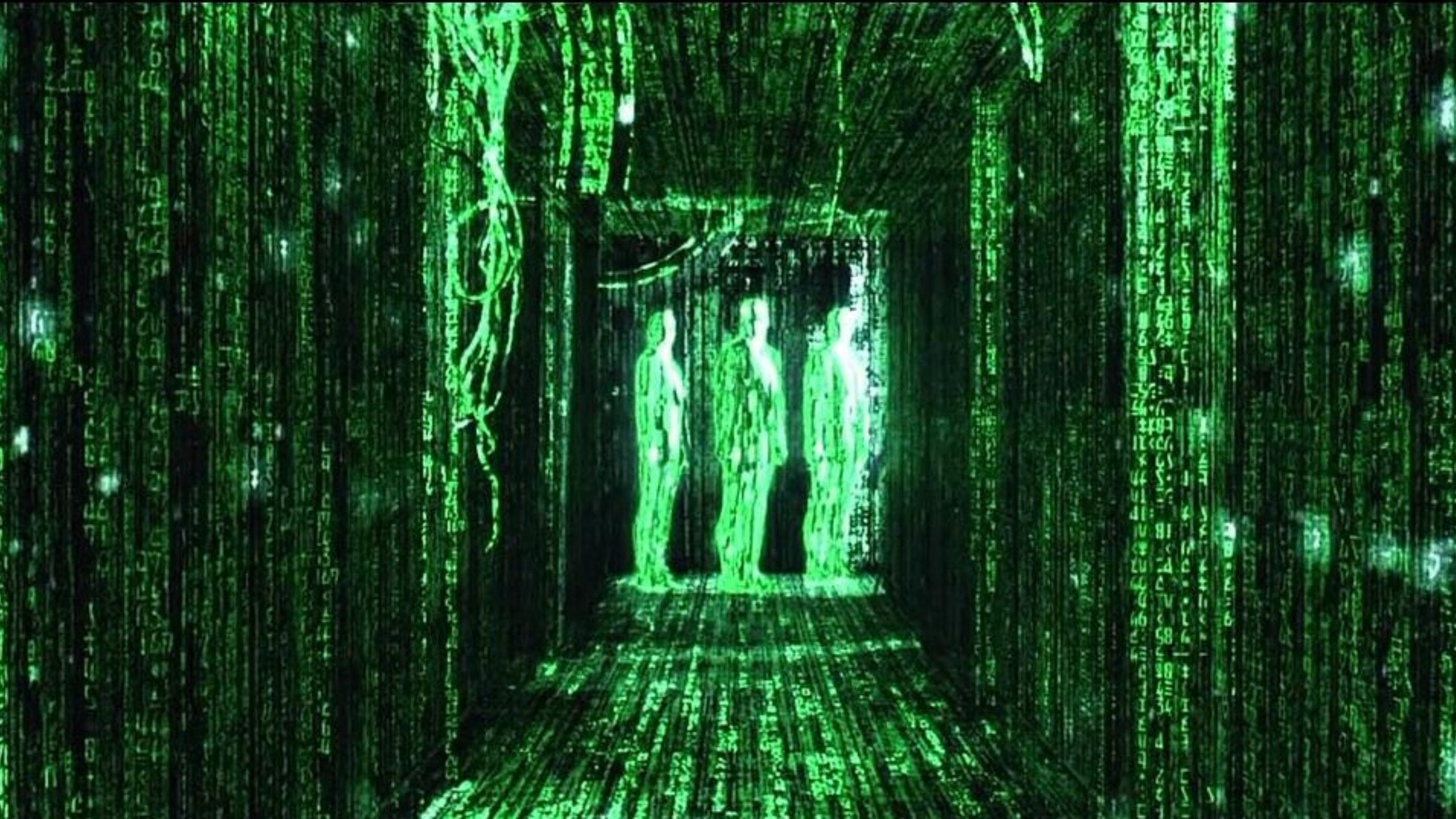


Q4U

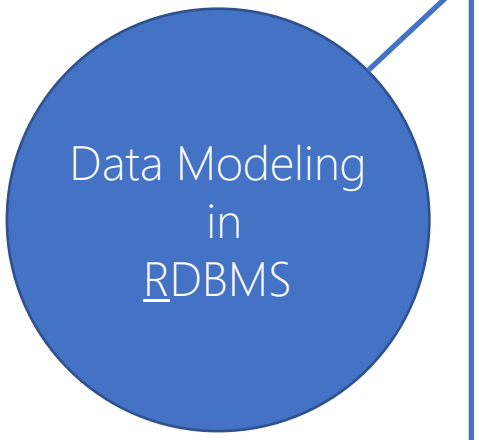
Given the following E/R, write relational expression to answer the following questions:

3- Which movie has won all Oscar awards? [Movie Title, Award Title, Year]





Today



Real World Entity

Conceptual Level | Entity-Relationship Model (E/R) Level

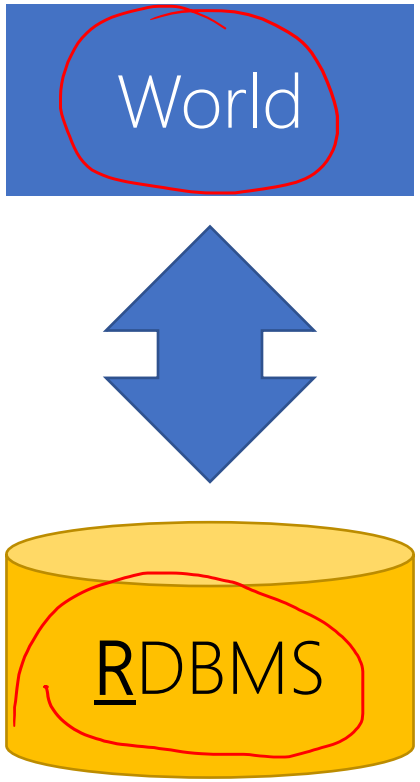
Conceptual Level | Logical Level | Relational Model

Conceptual Level | Logical Level | Physical Level | SQL

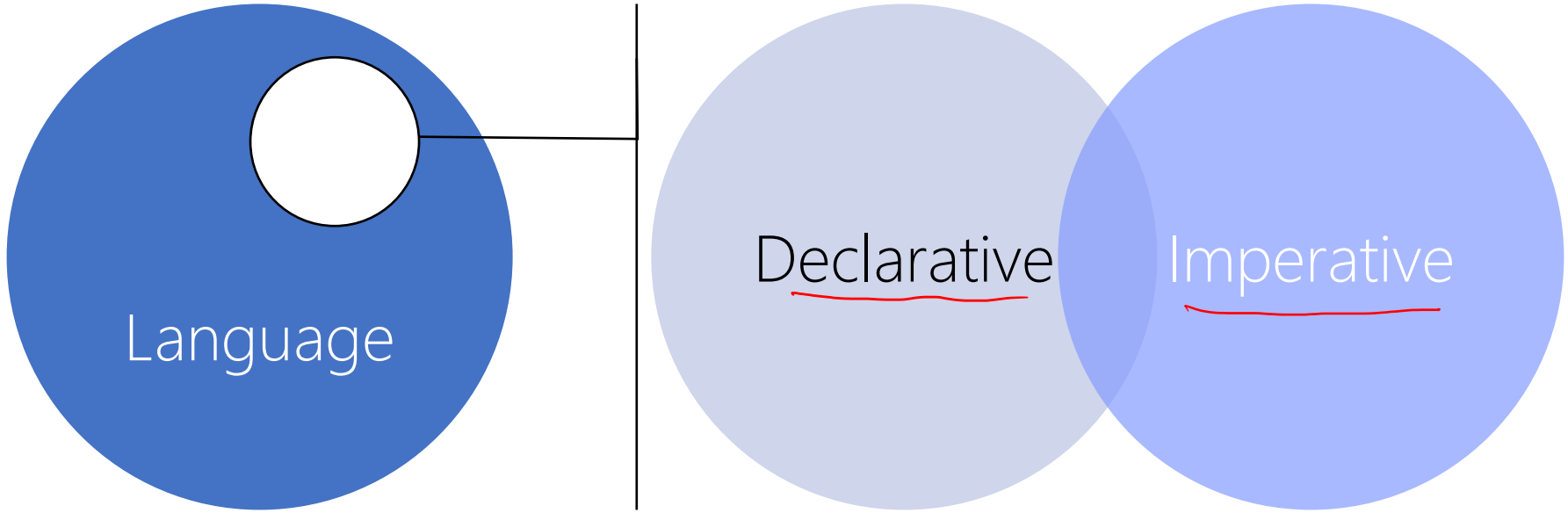
Conceptual Level | Logical Level | Computable Entity

SQL × Language

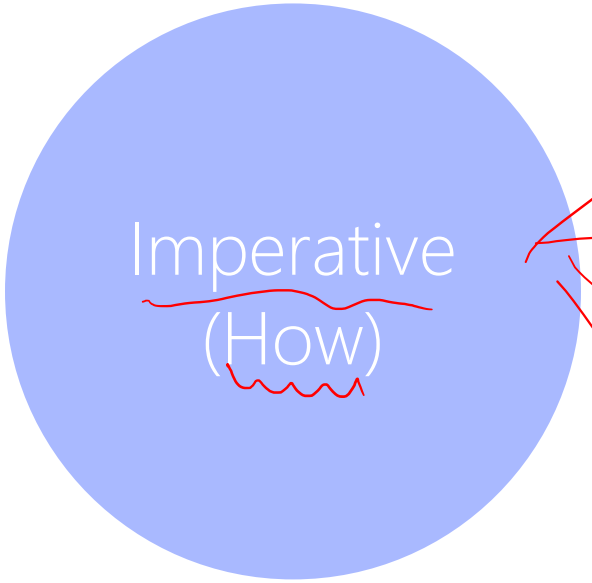
Communicate with Relational Database Management Systems (RDBMS)



Language × Types



Language × Types




```
#include <stdio.h>
void main() {
        int n;
        scanf("%d", &n)
        int r = 1;
        for(int i = n; i > 0; i--){
            r = r * i;
        }
        printf(r);
    }
}
```

Although correct, this program does not work in practice! (Why?)

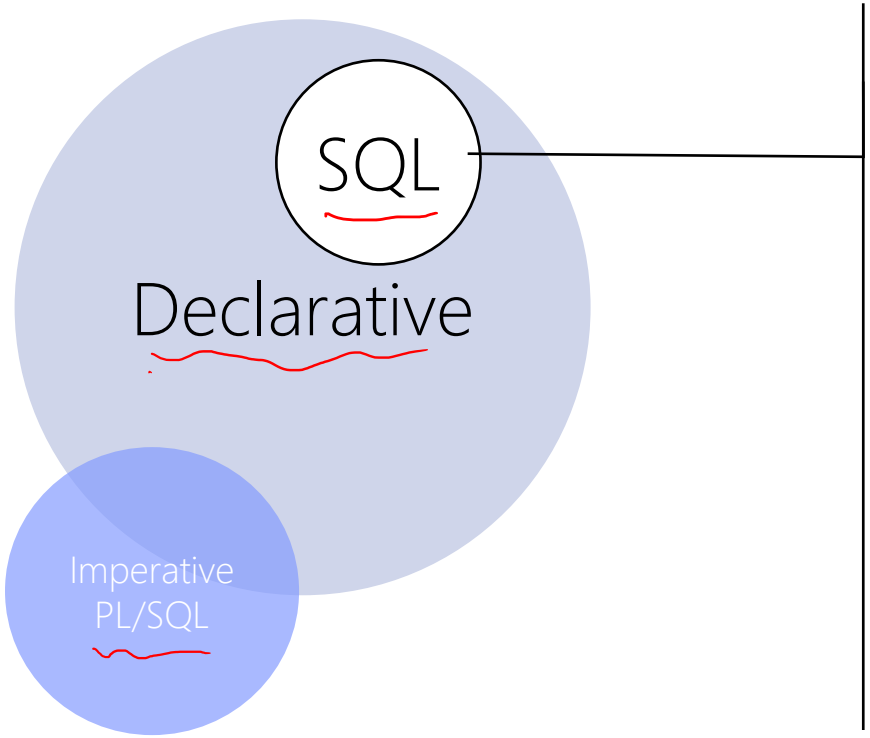
Language × Types

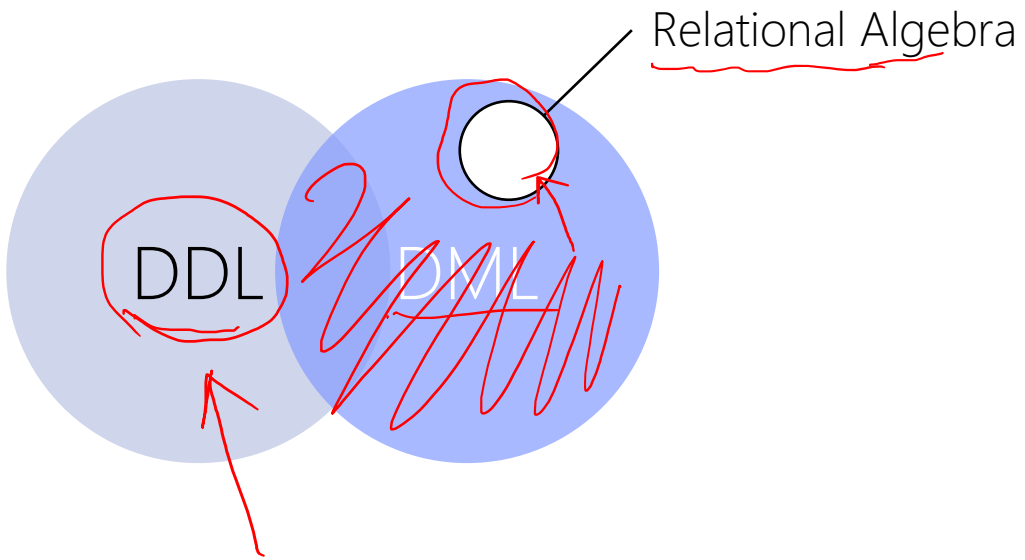
Declarative (What)

```
SELECT FACT(n);
```

The system figures out `best' way to execute query.

SQL × Declarative





SQL × Intro × History

The most commonly used relational DBMS's query and modify the database through a language called SQL (sometimes pronounced "sequel"). SQL stands for "Structured Query Language." The portion of SQL that supports queries has capabilities very close to that of relational algebra. However: SQL also includes statements for modifying the database (e.g., inserting and deleting tuples from relations) and for declaring a database schema. Thus, SQL serves as both a data-manipulation language and as a data definition language. SQL also standardizes many other database command. There are many different dialects of SQL. First, there are three major standards. There is ANSI (American National Standards Institute) SQL and an updated standard adopted in 1992, called SQL-92 or SQL2. The recent SQL-99 (previously referred to as SQL3) standard extends SQL2 with object-relational features and a number of other new capabilities. Then, there are versions of SQL produced by the principal DBMS vendors. These all include the capabilities of the original .ITS1 standard. They also conform to a large extent to the more recent SQL2, although each has its variations and extensions beyond SQL2, including some of the features in the SQL-99 standard. Herein, we consider SQL as a stand-alone query language.
SQLite does not fully support SQL-92.

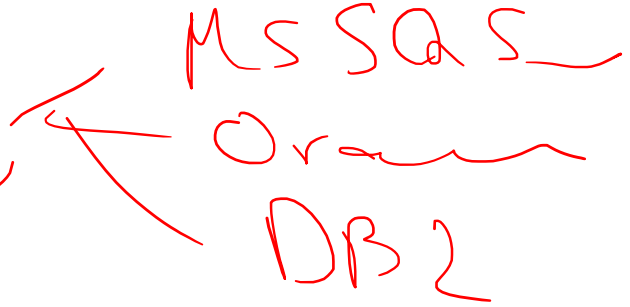
SQL × DDL

Data Definition Language to CREATE, ALTER, DROP relational

- Database
- Table

SQL × DDL × Database

CREATE DATABASE *DatabaseName;*



In SQLite> sqlite3 *DatabaseName;*

SQL × DDL × Database

DROP DATABASE *DatabaseName*;

In SQLite> simply delete database file

SQL × DDL × Database

ALTER DATABASE *DatabaseName* … ;

In SQLite> no command!

RENAME DATABASE *DatabaseName* TO *NewDatebaseName*

In SQLite> simply rename database file!

SQL × DDL × Database

Since there might be more than one database in a DBMS, we can switch between them by:

USE *DatabaseName*;

All remaining commands are done on *DatabaseName*

In SQLite> sqlite3 *DatabaseName*;

SQL × DDL × Table (Simple)


```
CREATE TABLE TableName(
        ColumnName1 DataType [NULL | NOT NULL],
        ColumnName2 DataType [NULL | NOT NULL],
        ...,
        ColumnNameN DataType [NULL | NOT NULL],
);
```

SQL × DDL × Table (Simple)


```
CREATE TABLE Movie(
        Title VARCHAR(255) NOT NULL,
        ReleaseDate DATE,
        Language VARCHAR(255),
        RunningTime INTEGER
);

CREATE TABLE Director(
        FirstName VARCHAR(255) NOT NULL,
        LastName VARCHAR(255) NOT NULL,
);
```

SQL × DDL × Table (Simple) × Datatype 15

INTEGER | INT
DECIMAL(i, j): `i' total #digits, `j' #digits after decimal point (.), e.g., 12.351 ∈ DECIMAL(5,3)
FLOAT | REAL: single precision real number
DOUBLE: double precision real number

DATE: year month day, the format can be modified, e.g., yy-mm-dd | mm-dd-yyyy
TIME: hour:minute:second, the format can be modified, precision depends on DBMS
DATETIME
TIMESTAMP

CHAR(n): `n' character, if less, padded with space
VARCHAR(n): `n' character max, if less, no padding
TEXT: document, article, ...

BIT: only one bit, 0 | 1
BOOLEAN: TRUE | FALSE
BLOB: large object in binary format (voice, movie, image, ...)

SQL × DDL × Table (Simple) × Datatype 16

Data types are highly dependent on the underlying DBMS. See manual of the DBMS.

e.g., SQLite → https://www.sqlite.org/datatype3.html

SQL × DDL × Table × Primary Key


```
CREATE TABLE TableName(
        ColumnName1 DataType PRIMARY KEY,
        ColumnName2 DataType,
        ...
        ColumnNameN DataType,
);
```

```
CREATE TABLE TableName(
        ColumnName1 DataType,
        ColumnName2 DataType,
        ...
        ColumnNameN DataType,
        CONSTRAINT PK_Name PRIMARY KEY(column names)
);
```

SQL × DDL × Table × Primary Key


```
CREATE TABLE Movie(
        Title VARCHAR(255) PRIMARY KEY,
        ReleaseDate DATE,
        Language VARCHAR(255),
        RunningTime INTEGER
);

CREATE TABLE Director(
        FirstName VARCHAR(255) NOT NULL,
        LastName VARCHAR(255) NOT NULL,
        CONSTRAINT PK_FirstName_LastName PRIMARY KEY(FirstName, LastName)
);
```

Could be any name, but by convention we follow this:
PK_ColumnName1_ColumnName2_...

SQL × DDL × Table × Primary Key


```
CREATE TABLE Movie(
        Title VARCHAR(255) PRIMARY KEY,
        ReleaseDate DATE,
        Language VARCHAR(255),
        RunningTime INTEGER,
        CONSTRAINT PK_Title PRIMARY KEY(Title)
);

CREATE TABLE Director(
        FirstName VARCHAR(255) NOT NULL,
        LastName VARCHAR(255) NOT NULL,
        CONSTRAINT PK_FirstName_LastName PRIMARY KEY(FirstName, LastName)
);
```

Could be any name, but by convention we follow this:
PK_ColumnName1_ColumnName2_...

SQL × DDL × Table × Surrogate Key 20

```
CREATE TABLE TableName(
        Id INTEGER PRIMARY KEY AUTOINCREMENT,
        ColumnName1 DataType,
        ColumnName2 DataType,
        ...
        ColumnNameN DataType,
);
```

SQL × DDL × Table × Candidate Key


```
CREATE TABLE TableName(
        ColumnName1 DataType [UNIQUE],
        ColumnName2 DataType [UNIQUE],
        ...
        ColumnNameN DataType [UNIQUE],
);

CREATE TABLE TableName(
        ColumnName1 DataType,
        ColumnName2 DataType,
        ...
        ColumnNameN DataType,
        CONSTRAINT UK_Name UNIQUE(column names)
);
```

Ck

SQL × DDL × Table × Candidate Key


```
CREATE TABLE Movie(
        Id INTEGER PRIMARY KEY AUTOINCREMENT,
        Title VARCHAR(255) UNIQUE,
        ReleaseDate DATE,
        Language VARCHAR(255),
        RunningTime INTEGER,
        CONSTRAINT UK_Title UNIQUE(Title)
);

CREATE TABLE Director(
        Id INTEGER PRIMARY KEY AUTOINCREMENT,
        FirstName VARCHAR(255) NOT NULL,
        LastName VARCHAR(255) NOT NULL,
        CONSTRAINT UK_FirstName_LastName UNIQUE(FirstName, LastName)
);
```

Could be any name, but by convention:
UK_ColumnName1_ColumnName2_...