# Chapter 1

# An introduction to relational databases and SQL

# Objectives

## Knowledge

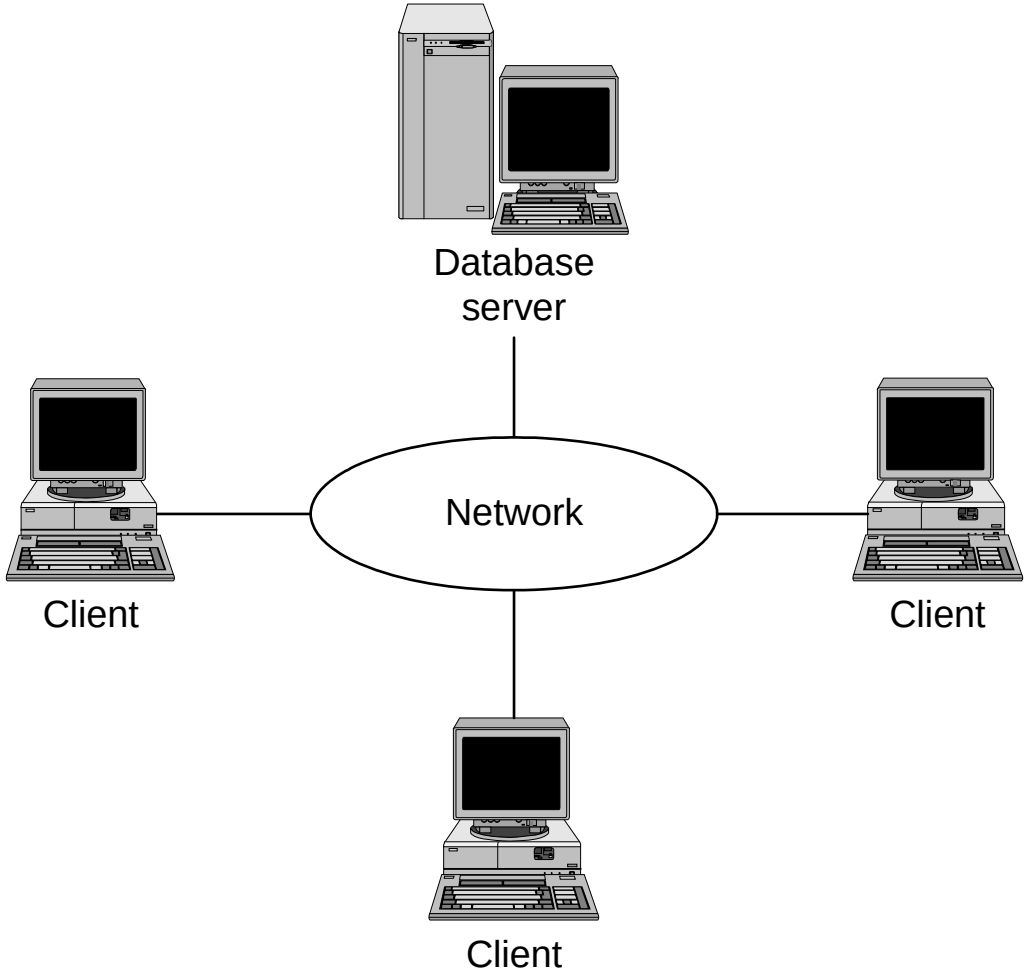
- Identify the three main hardware components of a client/server system.
- Describe the way a client accesses the database on a server using these terms: application software, data access API, database management system, SQL query, and query results.
- Describe the way a relational database is organized using these terms: tables, columns, rows, cells, primary keys, and foreign keys.
- Identify the three types of relationships that can exist between two tables.
- Describe the way the columns in a table are defined using these terms: data type, null value, and default value.
- Describe the difference between DML statements and DDL statements.

# Objectives (continued)

## Knowledge

- Describe the difference between an action query and a SELECT query.
- List three coding techniques that can make your SQL code easier to read and maintain.
- Explain how views and stored procedures differ from SQL statements that are issued from an application program.
- Describe the use of a database driver.

# A simple client/server system



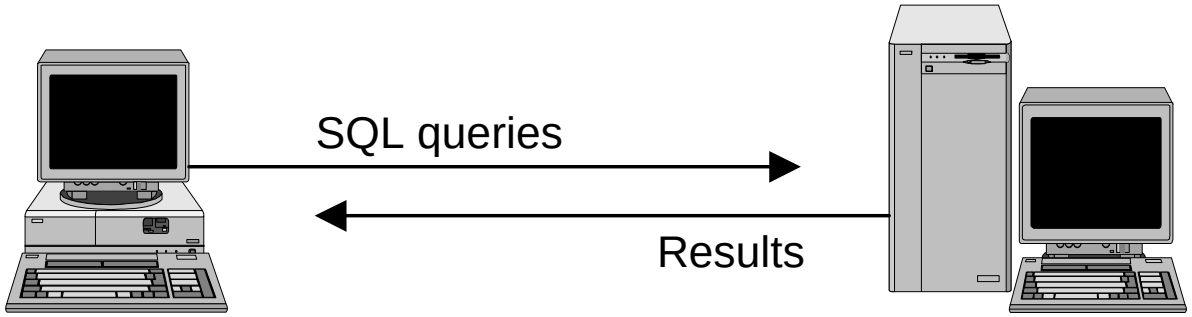
## The three hardware components of a client/server system

- Clients
- Server
- Network

## Terms to know

- Local area network (LAN)
- Wide area network (WAN)
- Enterprise system

# Client software, server software, and the SQL interface




**Client**
Application software
Data access API

**Database server**
Database management system
Database

## Server software

- Database management system (DBMS)
- The DBMS does the *back-end processing*

## Client software

- Application software
- Data access API (application programming interface)
- The client software does the *front-end processing*

## The SQL interface

- SQL queries
- *SQL* stands for *Structured Query Language*

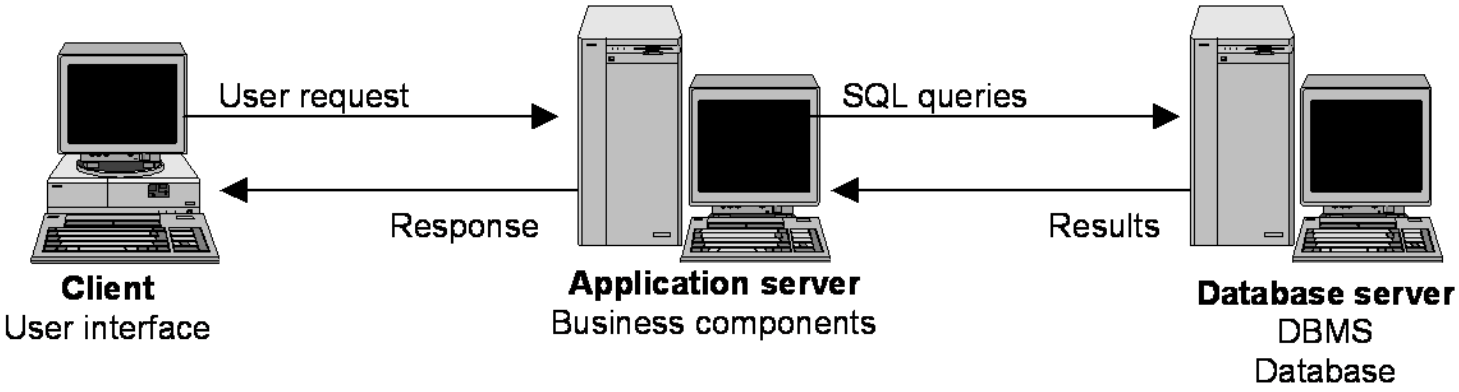
## Client/server system

- Processing is divided between client and server

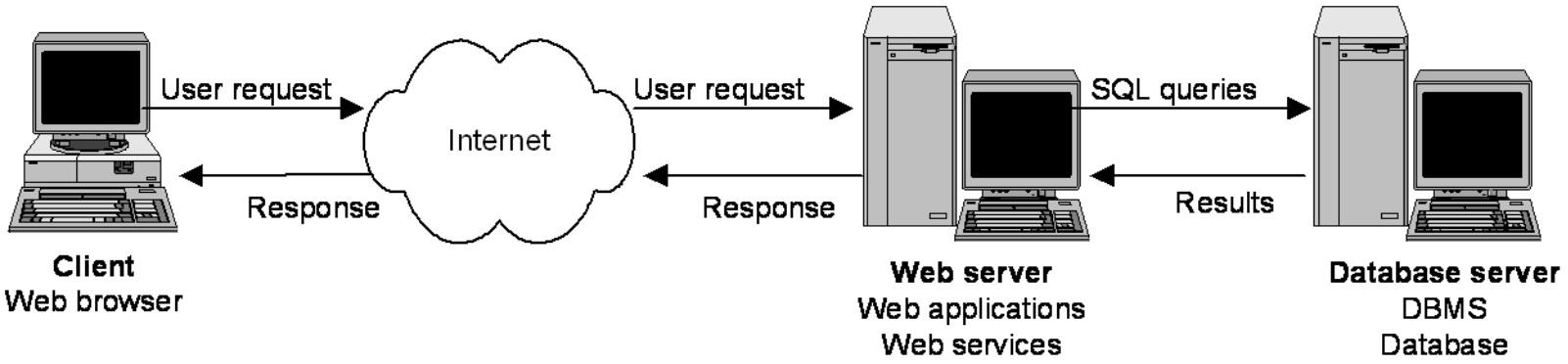
## File-handling system

- All processing is done by the clients

# An application that uses an application server



# A simple web-based system



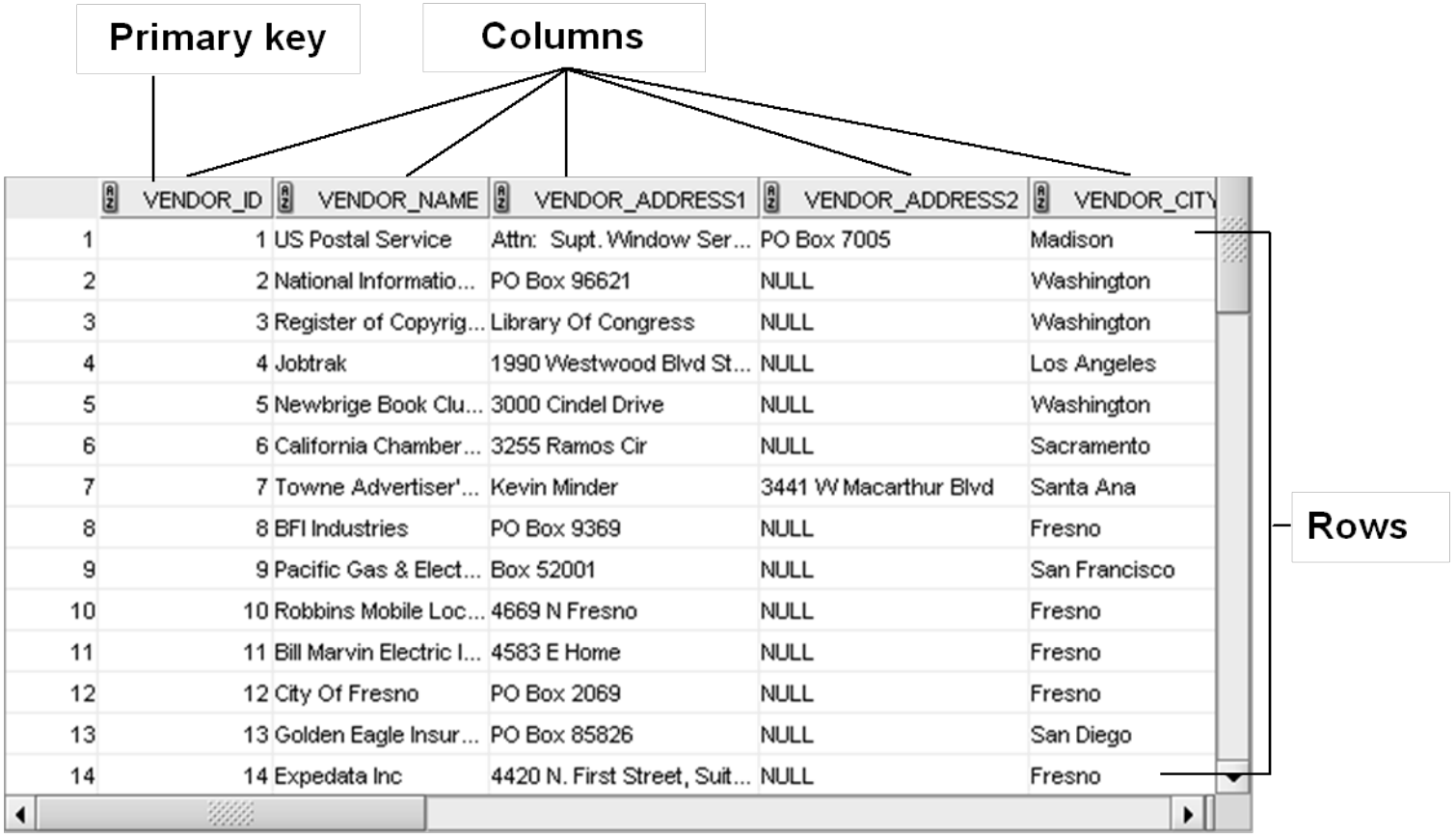
## Other client/server components

- Application servers store business components
- Web servers store web applications and web services

## How web applications work

- Web browser on a client sends a request to a web server.
- Web server processes the request.
- Web server passes any data requests to the database server.
- Database server returns results to web server.
- Web server returns a response to the browser.

# The Vendors table in an Accounts Payable database

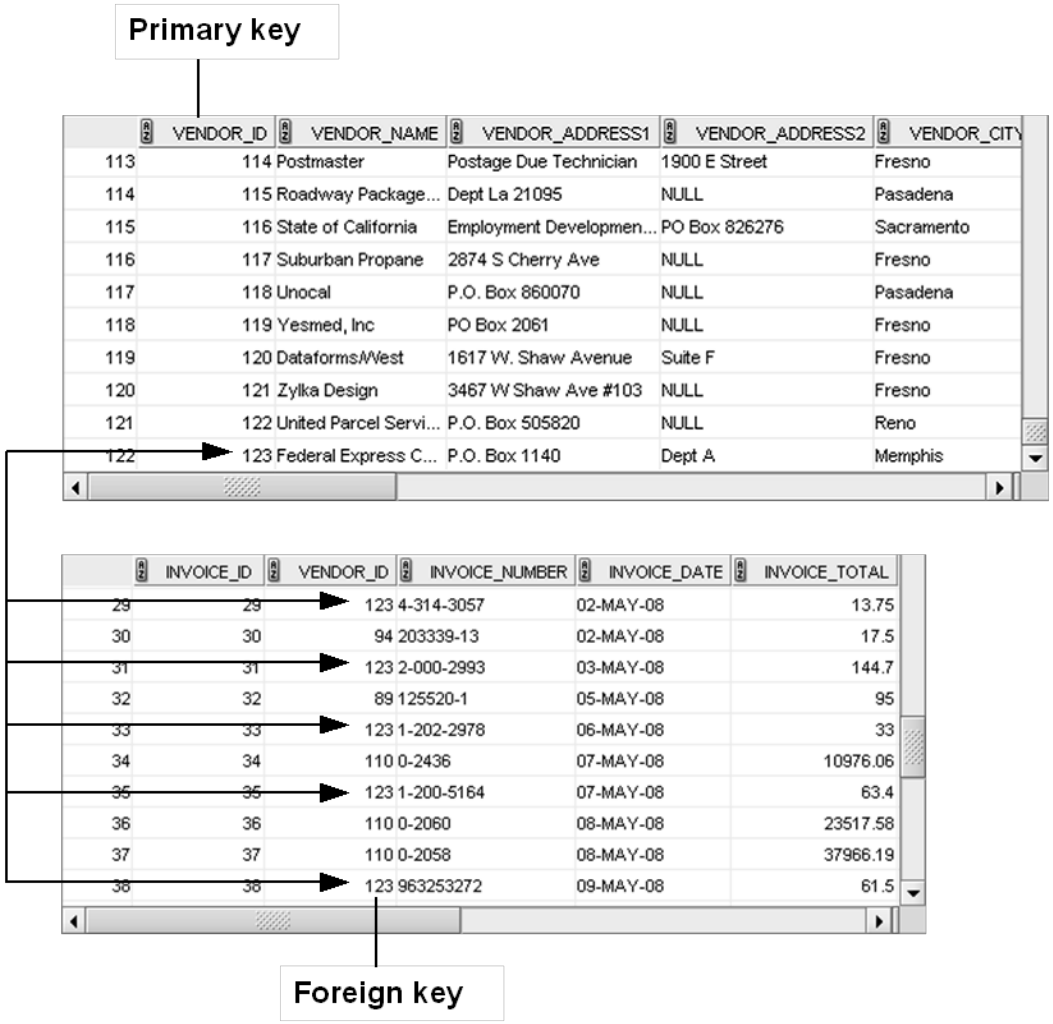



| | VENDOR_ID | VENDOR_NAME | VENDOR_ADDRESS1 | VENDOR_ADDRESS2 | VENDOR_CITY |
|---|---|---|---|---|---|
| 1 | 1 | US Postal Service | Attn: Supt. Window Ser... | PO Box 7005 | Madison |
| 2 | 2 | National Informatio... | PO Box 96621 | NULL | Washington |
| 3 | 3 | Register of Copyrig... | Library Of Congress | NULL | Washington |
| 4 | 4 | Jobtrak | 1990 Westwood Blvd St... | NULL | Los Angeles |
| 5 | 5 | Newbrige Book Clu... | 3000 Cindel Drive | NULL | Washington |
| 6 | 6 | California Chamber... | 3255 Ramos Cir | NULL | Sacramento |
| 7 | 7 | Towne Advertiser'... | Kevin Minder | 3441 W Macarthur Blvd | Santa Ana |
| 8 | 8 | BFI Industries | PO Box 9369 | NULL | Fresno |
| 9 | 9 | Pacific Gas & Elect... | Box 52001 | NULL | San Francisco |
| 10 | 10 | Robbins Mobile Loc... | 4669 N Fresno | NULL | Fresno |
| 11 | 11 | Bill Marvin Electric I... | 4583 E Home | NULL | Fresno |
| 12 | 12 | City Of Fresno | PO Box 2069 | NULL | Fresno |
| 13 | 13 | Golden Eagle Insur... | PO Box 85826 | NULL | San Diego |
| 14 | 14 | Expedata Inc | 4420 N. First Street, Suit... | NULL | Fresno |

## Terms to know

- Relational database
- Table
- Row
- Column
- Cell
- Primary key
- Composite primary key
- Non-primary key (unique key)
- Index

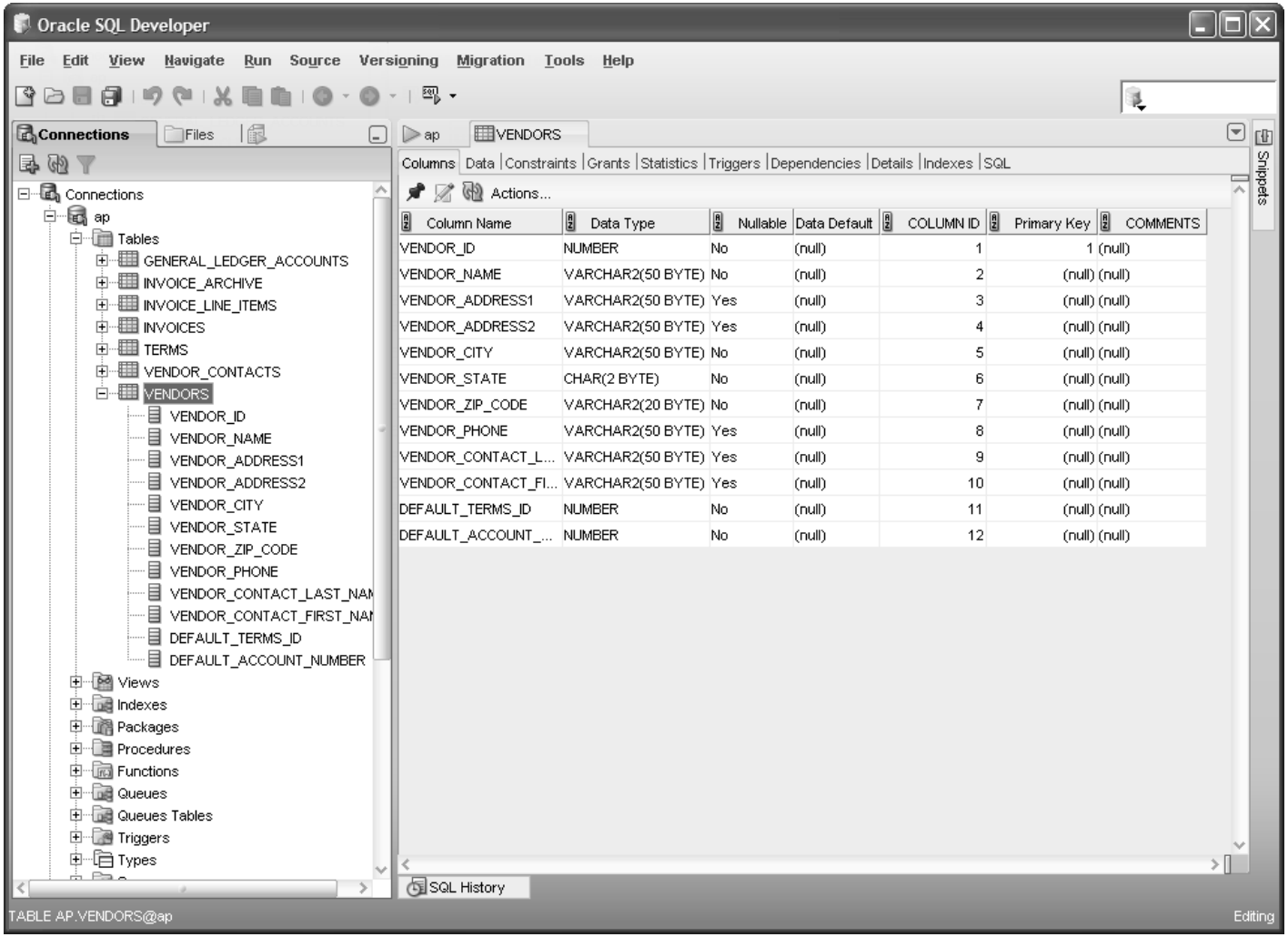
# The relationship between two tables



## Terms to know

- Foreign key
- One-to-many relationship
- One-to-one relationship
- Many-to-many relationship

# The columns of the Invoices table



## Common Oracle data types

- CHAR, VARCHAR2
- NUMBER
- FLOAT
- DATE

## Terms to know

- Data type
- Null value
- Default value

# Important events in the history of SQL

| Year | Event |
|---|---|
| 1970 | Dr. E. F. Codd develops the relational database model. |
| 1978 | IBM develops the predecessor to SQL. |
| 1979 | Relational Software, Inc. (later renamed Oracle) releases the first relational DBMS, Oracle. |
| 1982 | IBM releases their first RDBMS, SQL/DS (SQL/Data System). |
| 1985 | IBM released DB2 (Database 2). |
| 1987 | Microsoft releases SQL Server. |
| 1989 | ANSI publishes first SQL standards (ANSI/ISO SQL-89, or SQL1). |
| 1992 | ANSI revises standards (ANSI/ISO SQL-92, or SQL2). |
| 1999 | ANSI publishes SQL3 (ANSI/ISO SQL:1999). |
| 2003 | ANSI publishes SQL4 (ANSI/ISO SQL:2003). |

## How knowing “standard SQL” helps you

- Basic SQL statements are the same for all dialects.
- Once you know one dialect, you can easily learn others.

## How knowing “standard SQL” does not help you

- Any application requires changes when moved to another database.

## First database releases

| | |
|---|---|
| Oracle | 1979 |
| DB2 | 1985 |
| SQL Server | 1987 |

## Primary platforms

| | |
|---|---|
| Oracle | Unix<br>OS/390 and z/OS |
| DB2 | Unix<br>OS/390 and z/OS |
| SQL Server | Windows |

## SQL DML statements

- SELECT
- INSERT
- UPDATE
- DELETE

## SQL DDL statements

- CREATE USER, TABLE, SEQUENCE, INDEX
- ALTER USER, TABLE, SEQUENCE, INDEX
- DROP USER, TABLE, SEQUENCE, INDEX
- GRANT
- REVOKE

## A statement that creates a new user for a database

```
CREATE USER ap IDENTIFIED BY ap
```

## A statement that grants privileges to a user

```
GRANT ALL PRIVILEGES TO ap
```

# A statement that creates a new table

```
CREATE TABLE invoices
(
  invoice_id       NUMBER         NOT NULL,
  vendor_id        NUMBER         NOT NULL,
  invoice_number   VARCHAR2(50)   NOT NULL,
  invoice_date     DATE           NOT NULL,
  invoice_total    NUMBER(9,2)    NOT NULL,
  payment_total    NUMBER(9,2)                  DEFAULT 0,
  credit_total     NUMBER(9,2)                  DEFAULT 0,
  terms_id         NUMBER         NOT NULL,
  invoice_due_date DATE           NOT NULL,
  payment_date     DATE,
  CONSTRAINT invoices_pk
    PRIMARY KEY (invoice_id),
  CONSTRAINT invoices_fk_vendors
    FOREIGN KEY (vendor_id)
    REFERENCES vendors (vendor_id),
  CONSTRAINT invoices_fk_terms
    FOREIGN KEY (terms_id)
    REFERENCES terms (terms_id)
)
```

## A statement that adds a new column to a table

```
ALTER TABLE invoices
ADD balance_due NUMBER(9,2)
```

## A statement that deletes the new column

```
ALTER TABLE invoices
DROP COLUMN balance_due
```

## A statement that creates an index on the table

```
CREATE INDEX invoices_vendor_id_index
  ON invoices (vendor_id)
```

## A statement that deletes the new index

```
DROP INDEX invoices_vendor_id_index
```

# A statement that creates a sequence

```
CREATE SEQUENCE invoice_id_seq
START WITH 115
INCREMENT BY 1
```

# The Invoices base table

| | INVOICE_ID | VENDOR_ID | INVOICE_NUMBER | INVOICE_DATE | INVOICE_TOTAL | PAYMENT_TOTAL | CREDIT_TOTAL | TERMS_ID |
|---|---|---|---|---|---|---|---|---|
| 1 | 1 | 34 | QP58872 | 25-FEB-08 | 116.54 | 116.54 | 0 | 4 |
| 2 | 2 | 34 | Q545443 | 14-MAR-08 | 1083.58 | 1083.58 | 0 | 4 |
| 3 | 3 | 110 | P-0608 | 11-APR-08 | 20551.18 | 0 | 1200 | 5 |
| 4 | 4 | 110 | P-0259 | 16-APR-08 | 26881.4 | 26881.4 | 0 | 3 |
| 5 | 5 | 81 | MABO1489 | 16-APR-08 | 936.93 | 936.93 | 0 | 3 |
| 6 | 6 | 122 | 989319-497 | 17-APR-08 | 2312.2 | 0 | 0 | 4 |
| 7 | 7 | 82 | C73-24 | 17-APR-08 | 600 | 600 | 0 | 2 |
| 8 | 8 | 122 | 989319-487 | 18-APR-08 | 1927.54 | 0 | 0 | 4 |
| 9 | 9 | 122 | 989319-477 | 19-APR-08 | 2184.11 | 2184.11 | 0 | 4 |
| 10 | 10 | 122 | 989319-467 | 24-APR-08 | 2318.03 | 2318.03 | 0 | 4 |
| 11 | 11 | 122 | 989319-457 | 24-APR-08 | 3813.33 | 3813.33 | 0 | 3 |
| 12 | 12 | 122 | 989319-447 | 24-APR-08 | 3689.99 | 3689.99 | 0 | 3 |
| 13 | 13 | 122 | 989319-437 | 24-APR-08 | 2765.36 | 2765.36 | 0 | 2 |
| 14 | 14 | 122 | 989319-427 | 25-APR-08 | 2115.81 | 2115.81 | 0 | 1 |
| 15 | 15 | 121 | 97/553B | 26-APR-08 | 313.55 | 0 | 0 | 4 |

# A SELECT statement that retrieves and sorts selected rows

```
SELECT invoice_number, invoice_date, invoice_total,
    payment_total, credit_total,
    invoice_total - payment_total - credit_total AS
balance_due
FROM invoices
WHERE invoice_total - payment_total - credit_total > 0
ORDER BY invoice_date
```

# The result set for the SELECT statement

| | INVOICE_NUMBER | INVOICE_DATE | INVOICE_TOTAL | PAYMENT_TOTAL | CREDIT_TOTAL | BALANCE_DUE |
|---|---|---|---|---|---|---|
| 1 | P-0608 | 11-APR-08 | 20551.18 | 0 | 1200 | 19351.18 |
| 2 | 989319-497 | 17-APR-08 | 2312.2 | 0 | 0 | 2312.2 |
| 3 | 989319-487 | 18-APR-08 | 1927.54 | 0 | 0 | 1927.54 |
| 4 | 97/553B | 26-APR-08 | 313.55 | 0 | 0 | 313.55 |
| 5 | 97/553 | 27-APR-08 | 904.14 | 0 | 0 | 904.14 |
| 6 | 97/522 | 30-APR-08 | 1962.13 | 0 | 200 | 1762.13 |

# A SELECT statement that joins two tables

```
SELECT vendor_name, invoice_number, invoice_date,
invoice_total
FROM vendors INNER JOIN invoices
    ON vendors.vendor_id = invoices.vendor_id
WHERE invoice_total >= 500
ORDER BY vendor_name, invoice_total DESC
```

# The result set for the SELECT statement

| | VENDOR_NAME | INVOICE_NUMBER | INVOICE_DATE | INVOICE_TOTAL |
|---|---|---|---|---|
| 7 | Federal Express Corporation | 963253230 | 15-MAY-08 | 739.2 |
| 8 | Ford Motor Credit Company | 9982771 | 03-JUN-08 | 503.2 |
| 9 | Franchise Tax Board | RTR-72-3662-X | 04-JUN-08 | 1600 |
| 10 | Fresno County Tax Collector | P02-88D77S7 | 06-JUN-08 | 856.92 |
| 11 | IBM | Q545443 | 14-MAR-08 | 1083.58 |
| 12 | Ingram | 31359783 | 23-MAY-08 | 1575 |
| 13 | Ingram | 31361833 | 23-MAY-08 | 579.42 |
| 14 | Malloy Lithographing Inc | 0-2058 | 08-MAY-08 | 37966.19 |
| 15 | Malloy Lithographing Inc | P-0259 | 16-APR-08 | 26881.4 |
| 16 | Malloy Lithographing Inc | 0-2060 | 08-MAY-08 | 23517.58 |
| 17 | Malloy Lithographing Inc | P-0608 | 11-APR-08 | 20551.18 |
| 18 | Malloy Lithographing Inc | 0-2436 | 07-MAY-08 | 10976.06 |
| 19 | Pollstar | 77290 | 04-JUN-08 | 1750 |

# A statement that adds a row to the Invoices table

```
INSERT INTO invoices
  (invoice_id, vendor_id, invoice_number, invoice_date,
   invoice_total, terms_id, invoice_due_date)
VALUES
  (invoice_id_seq.NEXTVAL, 12, '3289175', '18-JUL-08',
   165, 3, '17-AUG-08')
```

## A statement that changes one value in one row

```
UPDATE invoices
SET credit_total = 35.89
WHERE invoice_number = '367447'
```

## A statement that changes one value in multiple rows

```
UPDATE invoices
SET invoice_due_date = invoice_due_date + 30
WHERE terms_id = 4
```

## A statement that deletes a selected invoice

```
DELETE FROM invoices
WHERE invoice_number = '4-342-8069'
```

## A statement that deletes all paid invoices

```
DELETE FROM invoices
WHERE invoice_total - payment_total - credit_total = 0
```

## Terms to know

- Query
- Result table (result set)
- Calculated value
- Join
- Inner join
- Outer join
- Action query

# A CREATE VIEW statement for a view

```
CREATE VIEW vendors_min AS
    SELECT vendor_name, vendor_state, vendor_phone
    FROM vendors
```

# The virtual table for the view

| | VENDOR_NAME | VENDOR_STATE | VENDOR_PHONE |
|---|---|---|---|
| 1 | US Postal Service | VM | (800) 555-1205 |
| 2 | National Information Data Ctr | DC | (301) 555-8950 |
| 3 | Register of Copyrights | DC | NULL |
| 4 | Jobtrak | CA | (800) 555-8725 |
| 5 | Newbrige Book Clubs | NJ | (800) 555-9980 |
| 6 | California Chamber Of Commerce | CA | (916) 555-6670 |
| 7 | Towne Advertiser's Mailing Svcs | CA | NULL |

# A SELECT statement that uses the view

```
SELECT *
FROM vendors_min
WHERE vendor_state = 'CA'
ORDER BY vendor_name
```

# The result set

| | VENDOR_NAME | VENDOR_STATE | VENDOR_PHONE |
|---|---|---|---|
| 1 | ASC Signs | CA | NULL |
| 2 | Abbey Office Furnishings | CA | (559) 555-8300 |
| 3 | American Express | CA | (800) 555-3344 |
| 4 | Aztek Label | CA | (714) 555-9000 |
| 5 | BFI Industries | CA | (559) 555-1551 |
| 6 | Bertelsmann Industry Svcs. Inc | CA | (805) 555-0584 |
| 7 | Bill Jones | CA | NULL |

## Terms to know

- View
- Virtual table

## A SELECT statement that's difficult to read

```
select invoice_number, invoice_date, invoice_total,
payment_total, credit_total, invoice_total - payment_total -
credit_total as balance_due from invoices where invoice_total
- payment_total - credit_total > 0 order by invoice_date
```

## A SELECT statement with a readable style

```
SELECT invoice_number, invoice_date, invoice_total,
    payment_total, credit_total,
    invoice_total - payment_total - credit_total
        AS balance_due
FROM invoices
WHERE invoice_total - payment_total - credit_total > 0
ORDER BY invoice_date
```

## SELECT statement with a block comment

```
/*
Author: Joel Murach
Date: 8/22/2008
*/
SELECT invoice_number, invoice_date, invoice_total,
    invoice_total - payment_total - credit_total
        AS balance_due
FROM invoices
```

## A SELECT statement with a single-line comment

```
-- The fourth column calculates the balance due
SELECT invoice_number, invoice_date, invoice_total,
    invoice_total - payment_total - credit_total
        AS balance_due
FROM invoices
```

## Coding recommendations

- Capitalize all keywords.
- Use lowercase for the other code.
- Separate the words in names with underscores.
- Start each clause on a new line.
- Break long clauses into multiple lines.
- Indent continued lines.
- Use comments only for code that is hard to understand.
- Make sure that the comments are correct and up-to-date.

# A CREATE PROCEDURE statement

```
CREATE OR REPLACE PROCEDURE update_invoices_credit_total
(
  invoice_number_param VARCHAR2,
  credit_total_param NUMBER
)
AS
BEGIN
  UPDATE invoices
  SET credit_total = credit_total_param
  WHERE invoice_number = invoice_number_param;

  COMMIT;
EXCEPTION
  WHEN OTHERS THEN
    ROLLBACK;
END;
/
```

# A statement that executes the stored procedure

```
CALL update_invoices_credit_total('367447', 35.89)
```

# A CREATE FUNCTION statement

```
CREATE OR REPLACE FUNCTION avg_invoice_total
(
   vendor_id_param INTEGER
)
RETURN NUMBER
AS
   avg_invoice_total_var NUMBER(9,2);
BEGIN
  SELECT AVG(invoice_total)
  INTO avg_invoice_total_var
  FROM invoices
  WHERE vendor_id = vendor_id_param;

  RETURN avg_invoice_total_var;
END;
/
```

# A statement that uses the function

```
SELECT vendor_id, invoice_total,
    avg_invoice_total(vendor_id)
FROM invoices
ORDER BY vendor_id
```

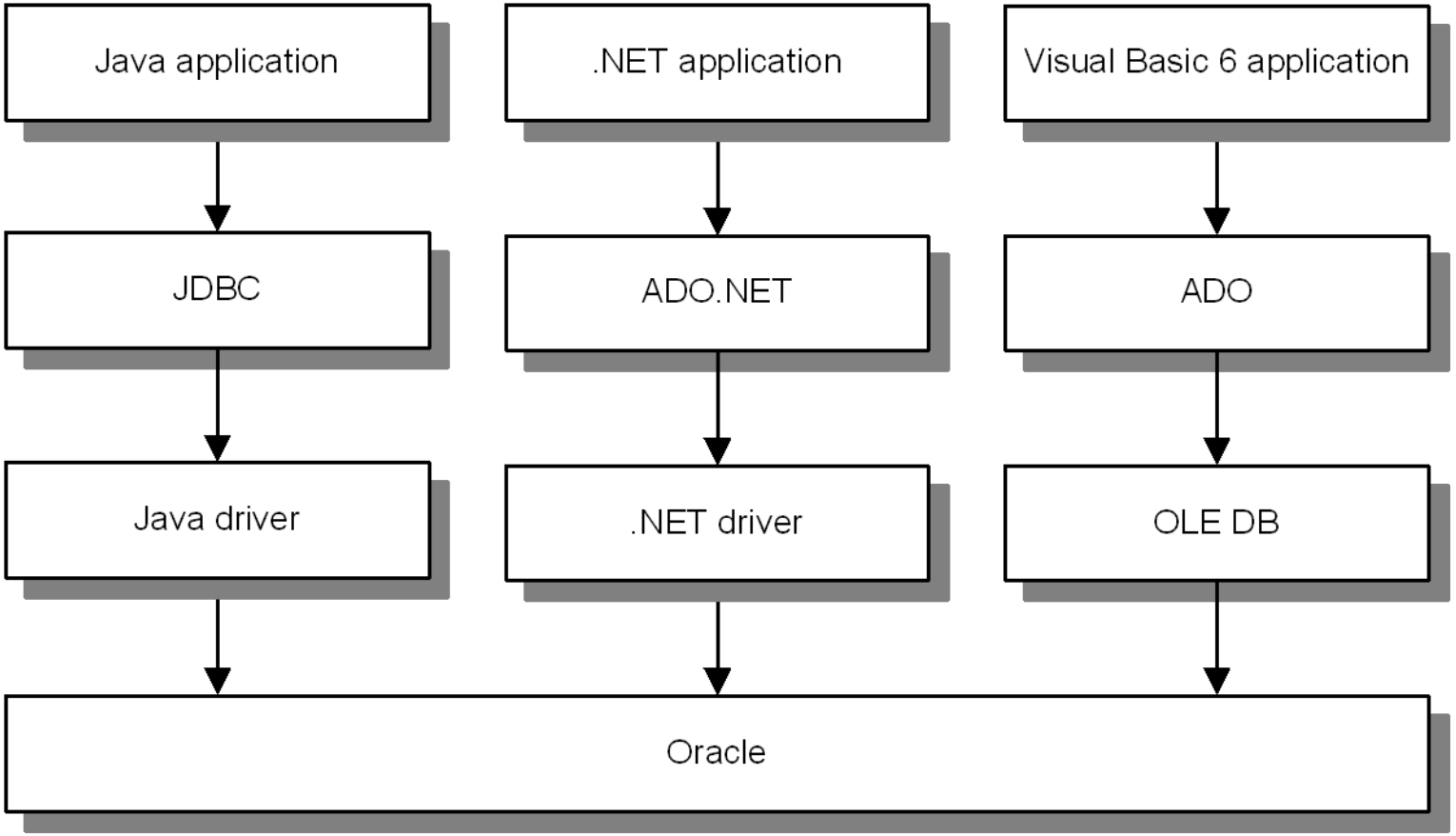
# The result set

| | VENDOR_ID | INVOICE_TOTAL | AVG_INVOICE_TOTAL(VENDOR_ID) |
|---|---|---|---|
| 1 | 34 | 1083.58 | 600.06 |
| 2 | 34 | 116.54 | 600.06 |
| 3 | 37 | 224 | 188 |
| 4 | 37 | 224 | 188 |
| 5 | 37 | 116 | 188 |
| 6 | 48 | 856.92 | 856.92 |
| 7 | 72 | 85.31 | 10963.66 |

## Terms to know

- PL/SQL (Procedure Language/SQL)
- Stored procedure
- Function (user-defined function or UDF)

# Common options for accessing Oracle data



## Two commonly used Oracle JDBC drivers

- Thin driver
- OCI driver

## Terms to know

- Data access model
- JDBC (Java Database Connectivity)
- ADO.NET (ActiveX Data Objects)
- Database driver

## The readme file for Oracle JDBC drivers

```
C:\oraclexe\app\oracle\product\10.2.0\server\jdbc\readme.txt
```

## The JAR file for the Oracle 10g Express JDBC driver

```
C:\oraclexe\app\oracle\product\10.2.0\server\jdbc\lib\
ojdbc14.jar
```

# A Java class that gets data from an Oracle database

```
import java.sql.*;
import java.text.NumberFormat;

public class DBTestApp
{
    public static void main(String args[])
    {
        // Load the database driver
        // NOTE: This block is for Oracle 10g (JDBC 3.0),
        // but not for Oracle 11g (JDBC 4.0)
        try
        {
            Class.forName("oracle.jdbc.OracleDriver");
        }
        catch(ClassNotFoundException e)
        {
            e.printStackTrace();
        }
```

# The Java class (continued)

```
// define common JDBC objects
Connection connection = null;
Statement statement = null;
ResultSet rs = null;
try
{
    // Connect to the database
    String dbUrl =
        "jdbc:oracle:thin:@localhost:1521:XE";
    String username = "ap";
    String password = "ap";
    connection = DriverManager.getConnection(
        dbUrl, username, password);
```

# The Java class (continued)

```
// Execute a SELECT statement
statement = connection.createStatement();
String query =
    "SELECT vendor_name, invoice_number,
         invoice_total " +
    "FROM vendors INNER JOIN invoices " +
    "    ON vendors.vendor_id =
            invoices.vendor_id " +
    "WHERE invoice_total >= 500 " +
    "ORDER BY vendor_name, invoice_total DESC";
rs = statement.executeQuery(query);
```

# The Java class (continued)

```
// Display the results of a SELECT statement
System.out.println(
    "Invoices with totals over 500:\n");
while(rs.next())
{
    String vendorName =
        rs.getString("vendor_name");
    String invoiceNumber =
        rs.getString("invoice_number");
    double invoiceTotal =
        rs.getDouble("invoice_total");

    NumberFormat currency =
        NumberFormat.getCurrencyInstance();
    String invoiceTotalString =
        currency.format(invoiceTotal);
```

# The Java class (continued)

```
                System.out.println(
                    "Vendor:     " + vendorName + "\n" +
                    "Invoice No: " + invoiceNumber + "\n" +
                    "Total:      " + invoiceTotalString +
                    "\n");
            }
        }
        catch(SQLException e)
        {
            e.printStackTrace();
        }
```

# The Java class (continued)

```
        finally
        {
            try
            {
                if (rs != null)
                    rs.close();
                if (statement != null)
                    statement.close();
                if (connection != null)
                    connection.close();
            }
            catch(SQLException e)
            {
                e.printStackTrace();
            }
        }
    }
}
```