IS475/675 Agenda: April 21, 2025

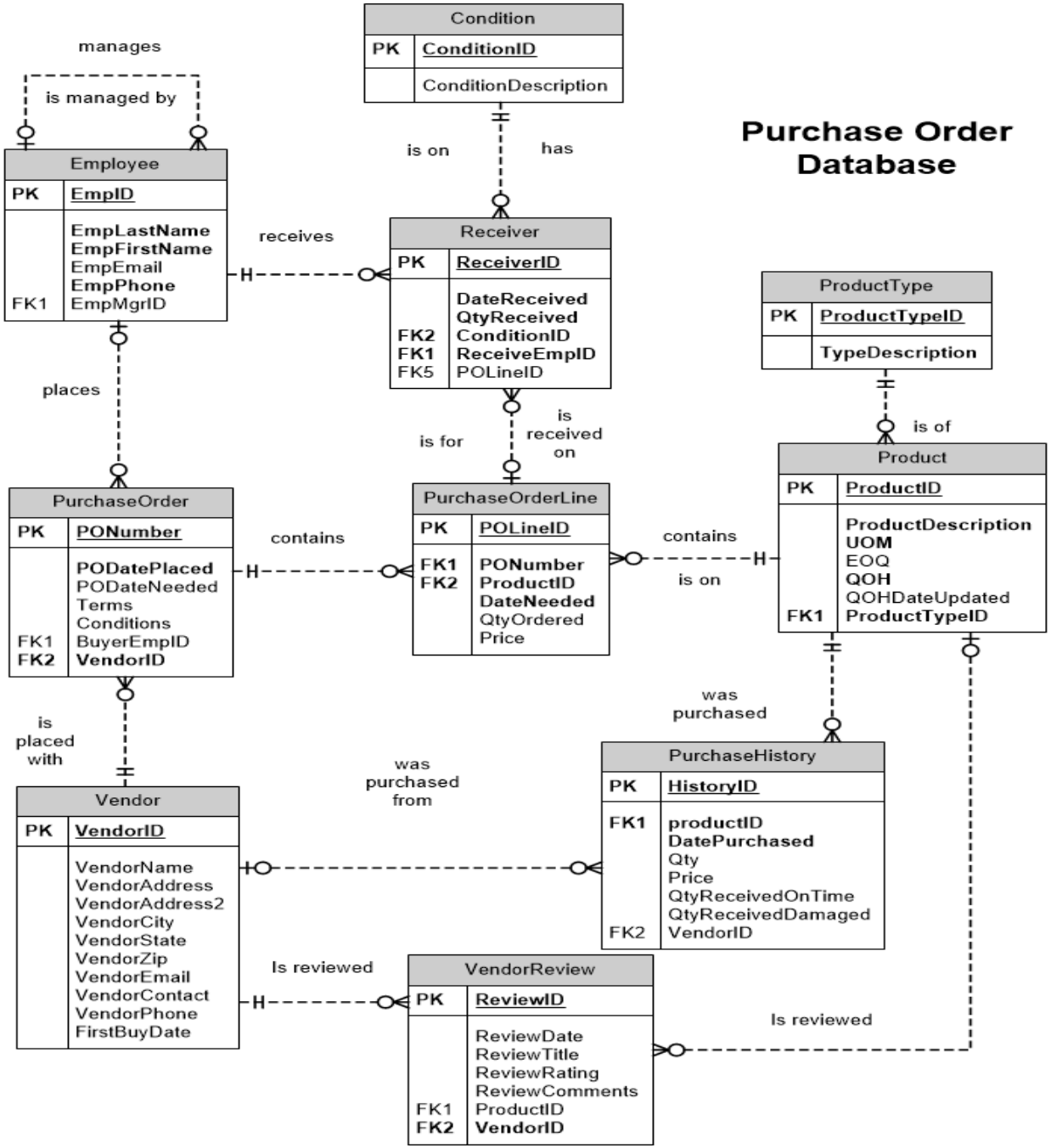
Review answers to HW#8 if requested.

Discuss HW#9 & 10 and explain how SQL Lab #9 supports those assignments.

Answer questions.

Define application partitioning and present how database programming fits within a partitioned environment.

Describe features of a DBMS that support partitioned applications.


Purchase Order Database
Condition
PK ConditionID
ConditionDescription
manages
is managed by
is on
has
Employee
PK EmpID
EmpLastName
EmpFirstName
EmpEmail
EmpPhone
FK1 EmpMgrID
receives
Receiver
PK ReceiverID
DateReceived
QtyReceived
FK2 ConditionID
FK1 ReceiveEmpID
FK5 POLineID
ProductType
PK ProductTypeID
TypeDescription
is of
places
is for
is received on
Product
PK ProductID
ProductDescription
UOM
EOQ
QOH
QOHDateUpdated
FK1 ProductTypeID
PurchaseOrder
PK PONumber
PODatePlaced
PODateNeeded
Terms
Conditions
FK1 BuyerEmpID
FK2 VendorID
contains
PurchaseOrderLine
PK POLineID
FK1 PONumber
FK2 ProductID
DateNeeded
QtyOrdered
Price
contains
is on
was purchased
is placed with
was purchased from
PurchaseHistory
PK HistoryID
FK1 productID
DatePurchased
Qty
Price
QtyReceivedOnTime
QtyReceivedDamaged
FK2 VendorID
Vendor
PK VendorID
VendorName
VendorAddress
VendorAddress2
VendorCity
VendorState
VendorZip
VendorEmail
VendorContact
VendorPhone
FirstBuyDate
Is reviewed
VendorReview
PK ReviewID
ReviewDate
ReviewTitle
ReviewRating
ReviewComments
FK1 ProductID
FK2 VendorID
Is reviewed

HW#9 – IS475 Two enhancements

- Requirement 1 – Accounts payable
  - Invoices received
  - Payments made
  - Transactions recorded
- Requirement 2 – Work Order
  - A product is used to make other products. A product is composed of other products. This is a many-to-many unary relationship with product.
  - A work order is a way to show which products are required to create other products. A work order is the intersection entity for the many-to-many unary relationship
  - A work order can have many work order lines.

HW#9 – IS 675 - Third enhancement

- More historical data.
  - More detailed information about closed purchase orders, purchase order lines, and receivers.
  - Suggestions from you as a database designer for additional historical data that might be helpful for business decision making.

HW#10 – IS 475

- Implementation of some of the changes.
- Will add new tables.
- Will add new data to existing tables.
- May update data in existing tables.
- May delete data from existing tables.

HW#10 – IS 675

- Must do the same enhancements as that for IS 475.
- Then has the option to:
  1. Create 2 each functions and stored procedures; or
  2. Create and populate the tables necessary to support the historical decision making designed in HW#9.
     - If selecting this option then must also write five queries to demonstrate the effectiveness of the historical decision making.

How to get data into a database?

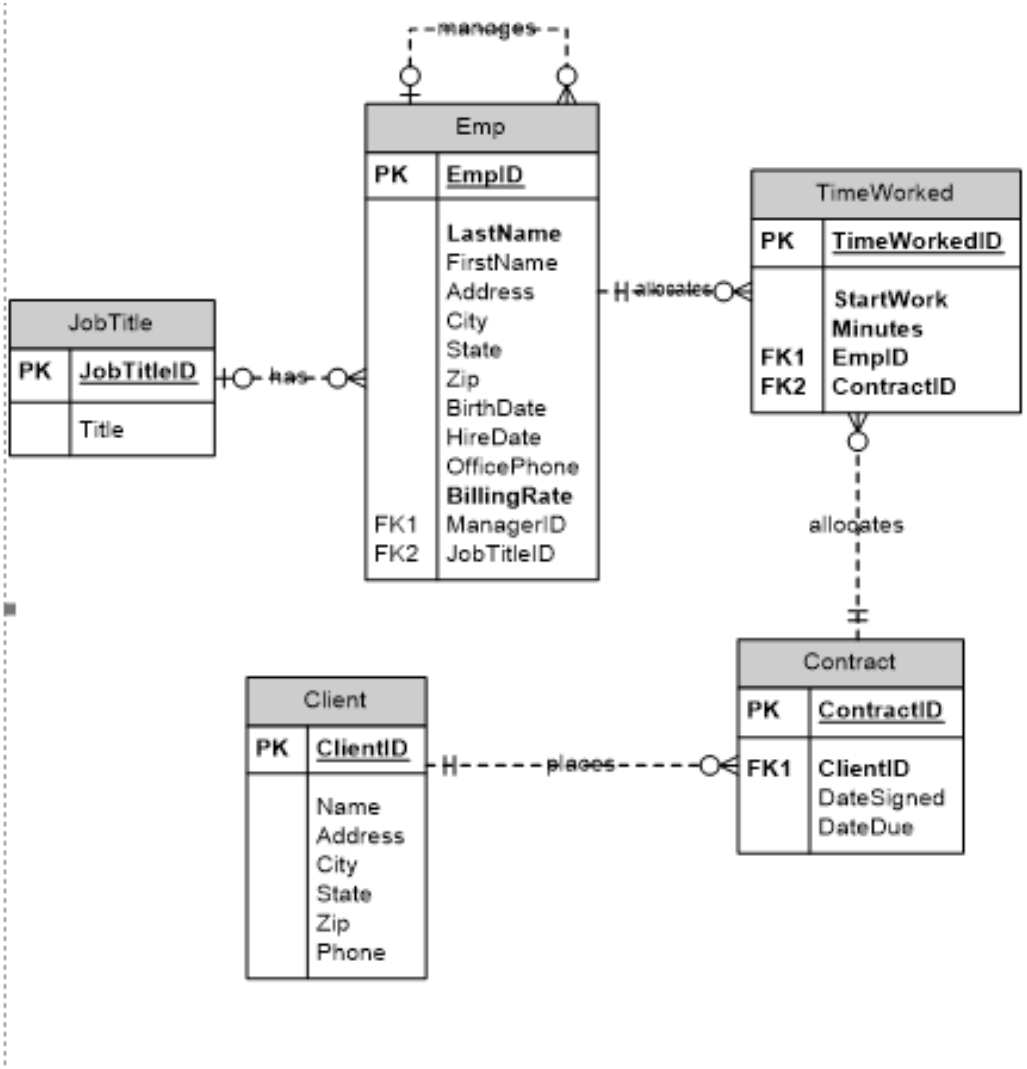
- Data must be accurate, consistent, and align with the structure of a relational database.
- This requires knowledge of the relative accuracy of the data and the structure of the relational database.
- Can do this at the data source.
- Can do this once the data is loaded into a table in a database.

SQL Lab Exercise #9 - ETL

- The goal of the lab is to add new data to existing tables.
- Here is the process:

1) Make and populate new tables – "y" tables – with a script file.
2) Make backup copies of those tables in case there are problems during the new data add process. We called these backup tables our "y14" tables.
3) Upload an ugly worksheet to a single table in SQL server called "alldata". This is a **load** process and you will use the SQL Server Import/Export wizard.
4) Using SQL, **extract** data from the "alldata" table into temporary tables ("el" tables), **transforming** the data into a format acceptable to our existing tables.
5) **Load** the data from the "el" table into the existing table ("y" table) one table at a time.

What tables are used for SQL Lab Exercise 9



What new SQL code did we learn in last week?

Examples of SQL Code used to "bulk load" data

```
INSERT INTO xJobTitle (jobtitleid, title)
SELECT      jobtitleid,
            title
FROM        alldata
```

This syntax requires that a table be created prior to executing the INSERT statement. The SELECT can include a WHERE clause to determine which rows to insert.

```
SELECT      jobtitleid,
            title
INTO        xJobTitle
FROM        alldata
```

This syntax creates a table and inserts the data in a single statement. The SELECT can also include a WHERE clause to determine which rows to insert.

What is application partitioning?

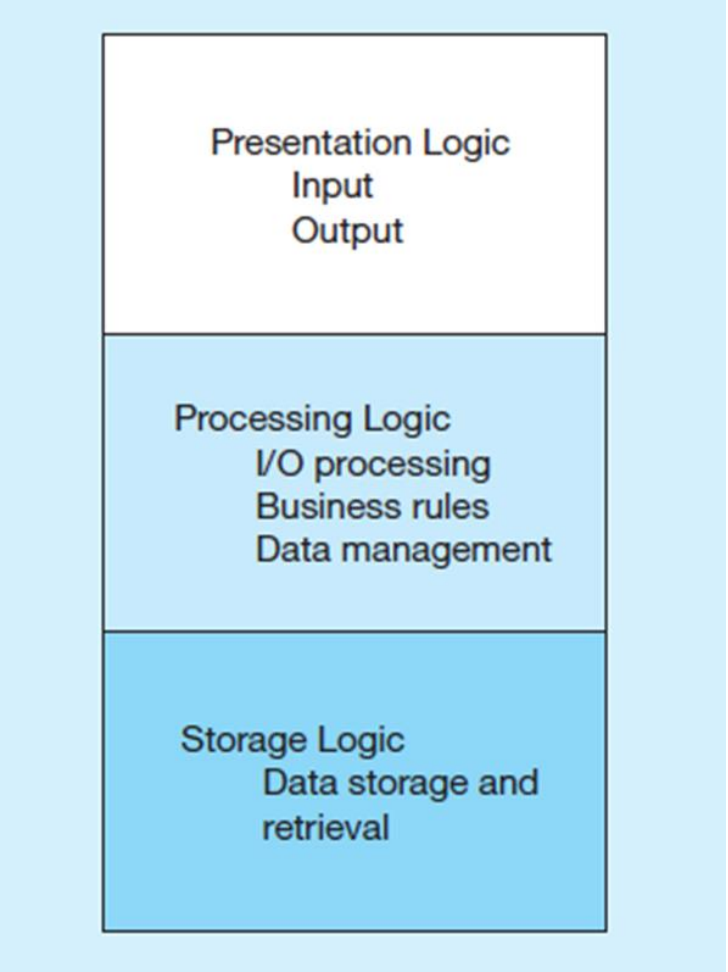
- Process of assigning portions of application code to client or server partitions.
- Described in Chapters 7 and 8 of the text "Modern Database Management"

Application Logic Components Depicted in Textbook

GUI interface (e.g. through a browser) that is accessed by the user of an application.

Procedures, functions, programs used to accomplish tasks within the application.

DBMS activities including inserting, updating, and deleting data within a database.



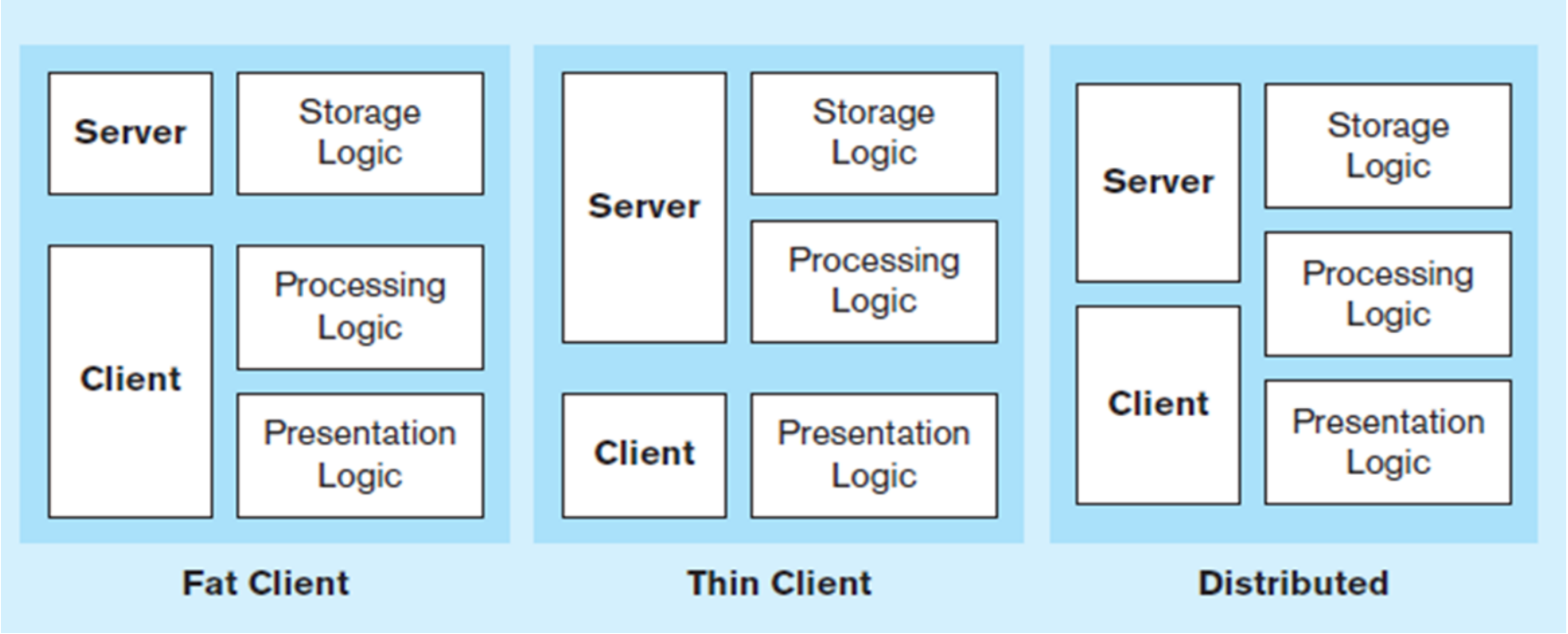
What are the goals of partitioning applications?

- Protect the integrity of the data.
- Secure the environment against unwanted intrusion.
- Provide good transaction throughput.
- Support sub-second response time.
- Make best use of available resources.

Common Logic Distributions – 2 Tier

Two-tier client-server environments

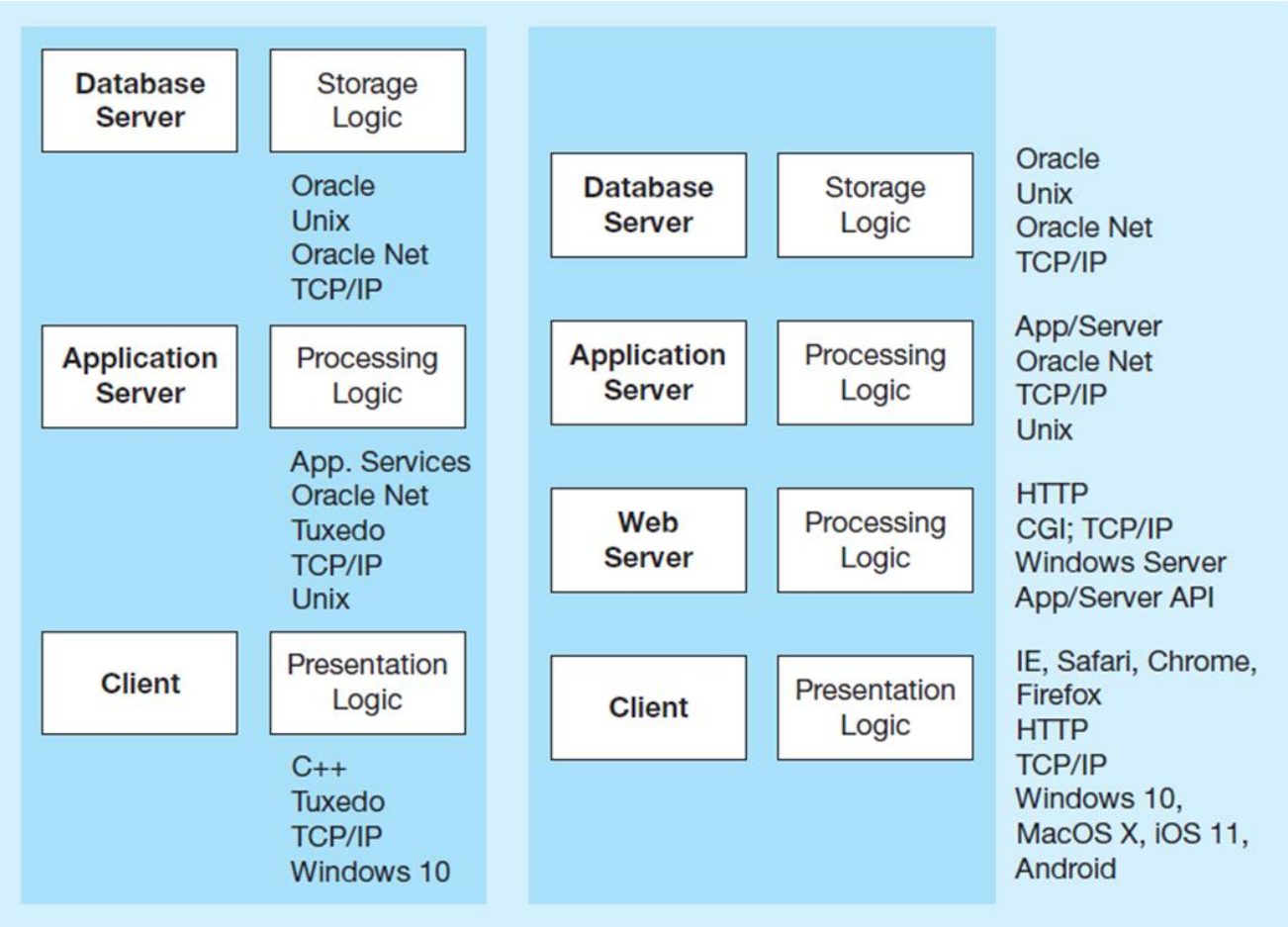
Processing logic could be at client (fat client), server (thin client), or both (distributed environment).



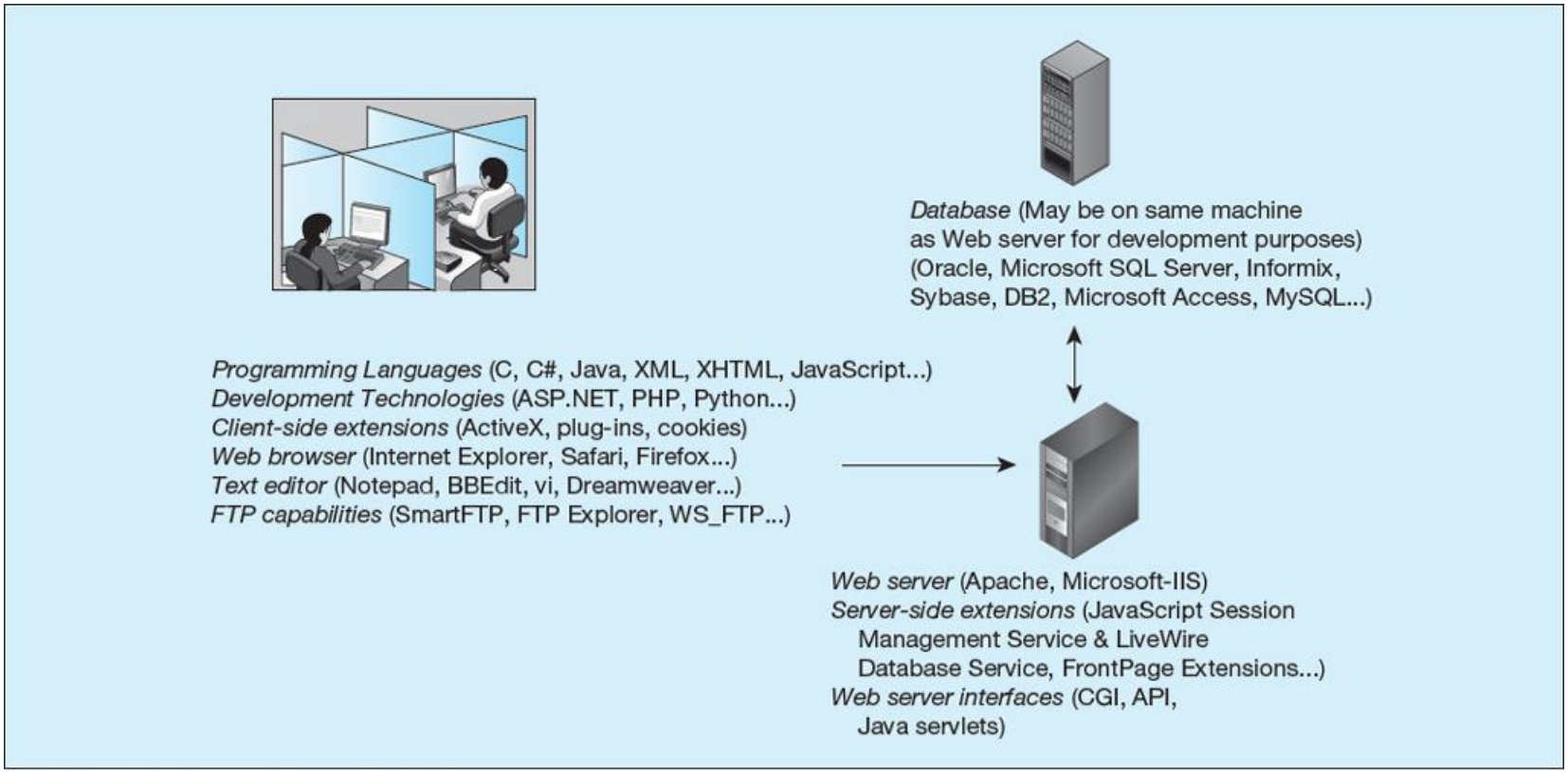
Common Logic Distributions – 3 Tier

Three-tier and *n*-tier client-server environments

Processing logic will be at application server or Web server.



Examples of products



Middleware and APIs

Middleware – software that allows an application to interoperate with other software without requiring user to understand and code low-level operations.

Application Programming Interface (API) – routines that an application uses to direct the performance of procedures. API's adhere to standards that are defined by computer vendors or standardizing bodies such as ANSI.

Common database APIs – ODBC, ADO.NET, JDBC

Steps for Using Databases via Middleware/APIs

1. Identify and register a database driver.
2. Open a connection to a database.
3. Execute a query against the database.
4. Process the results of the query.
5. Repeat steps 3–4 as necessary.
6. Close the connection to the database.

Database Access from a Java Program

```
import java.sql.*;
public class TestJDBC {
  public static void main(String[ ] args) {
  try {
  Driverd =
  (Driver)Class.forName("oracle.jdbc.driver.OracleDriver").newInstance( );
  System.out.println(d);
  DriverManager.registerDriver (new oracle.jdbc.driver.OracleDriver( ));
  Connection conn =
  DriverManager.getConnection ("jdbc:oracle:thin:@durga.uits.indiana.edu:1
  521:OED1", args[0], args[1]);


  Statement st = conn.createStatement( );
  ResultSet rec = st.executeQuery("SELECT * FROM Student");
  while(rec.next( )) {
    System.out.println(rec.getString("name"));}
  conn.close( );
    }
  catch (Exception e) {
System.out.println("Error – " + e);
  }
  }
}
```

Register the driver to be used.

Identify the type of driver to be used.

Open a connection to a database.

Create a Statement variable that can be used to issue queries against the database

Issue a query and get a result.

Process the result, one row at a time.

Close the connection.

Many examples and tutorials on the web

https://docs.microsoft.com/en-us/dotnet/api/system.data.sqlclient.sqlconnection.connectionstring?view=dotnet-plat-ext-5.0

https://www.guru99.com/c-sharp-access-database.html

https://www.astera.com/knowledge-center/python-to-sql-server-integration/

https://support.microsoft.com/en-us/help/308656/how-to-open-a-sql-server-database-by-using-the-sql-server-net-data-pro

https://www.connectionstrings.com/sql-server/

https://www.youtube.com/watch?v=g69lFxZdcVQ

Pop Quiz!

Which of the following is the process of assigning pieces of application code to clients or servers?

A. Application partitioning
B. Modularizing programs
C. Code distribution
D. Program breakup

Which of the following is **not** a basic step to accessing a database from an application:

A. register database driver.
B. open a connection.
C. define physical storage characteristics.
D. query the database.

In a **thin** client client/server 2-tier logic distribution:

A. The storage logic and the processing logic are resident on the client rather than the server.
B. The presentation logic is on the client and the processing and storage logic are on the server.
C. The storage logic is on the server and the processing and presentation logic are on the client.
D. The storage, processing and presentation logic are on the server.

Why focus on database?

- Stored data is fundamental to all business and organizational systems.
- Database programming can be used to:
  - Transform data from one form to another.
  - Automate related processes.
  - Help programmers working within other layers have a standard way to access and update the database.
  - Improve database processing speed.
  - Protect the integrity of data.

Four different types of procedural SQL programs

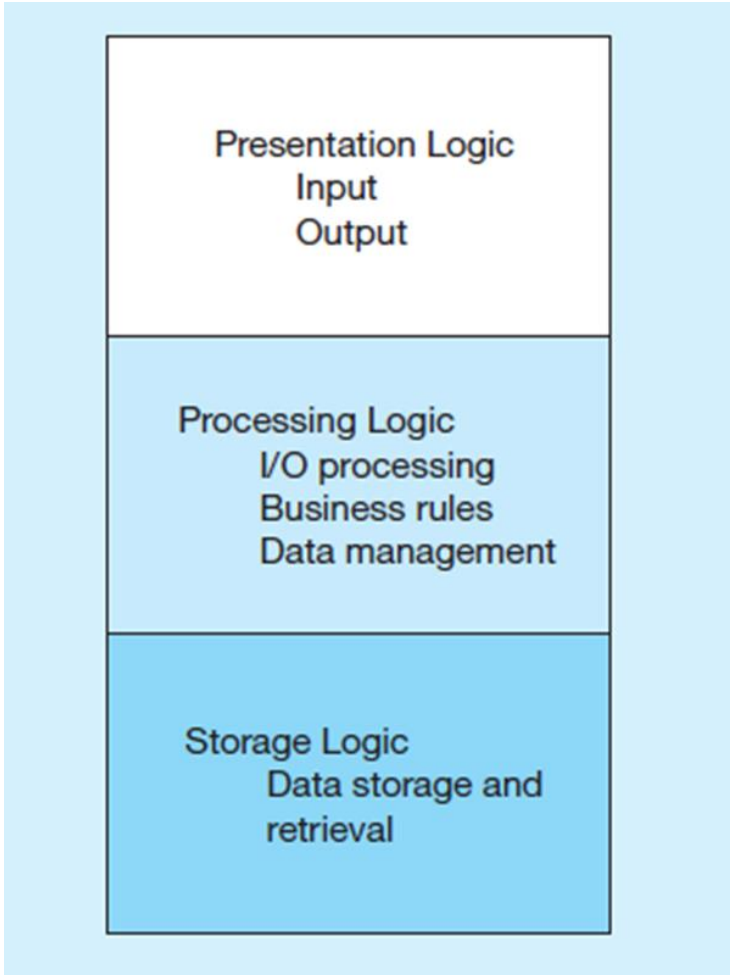
Type	Purpose	How it's usually executed	Where stored	Accepts parameters/ arguments?
Script	Execute multiple SQL statements	Within a client tool such as Management Studio	In a file on disk	No
Function	Automate a specific task	By an application or a SQL script	Database object	Yes
Stored Procedure	Execute multiple SQL statements in a pre-compiled format. Improve automation and efficiency.	By an application, by a person, or by a pre-specified time.	Database object	Yes
Trigger	Specialized stored procedure related to INSERT, UPDATE or DELETE.	Automatically by the database server when a specific transaction occurs.	Database object	No

Layered Software Architecture – with Database Programming

Functions

Stored Procedures

Triggers



What is the role of database (server-based) programming in application partitioning?

- Functions
  - Embedded within the SQL code, can help facilitate more flexible processing within all layers.
- Stored procedures
  - Code logic embedded in the DBMS.
  - Improves performance of application processing.
- Triggers
  - Can help protect the integrity of transactions.

User-defined functions

- Program code that accepts parameters, performs an action, and returns the result of the action.
- A function differs from a stored procedure:
  - Is designed to return a value (either scalar or a table);
  - Can be executed directly from a SELECT statement.
- Most often used to create values that will be used in many different queries.

Examples of use (functions)

- Do a calculation, such as age, BMI, present value, or median.
- Do customized formatting, such as formatting a social security number or a U.S. telephone number.
- Do validation, such as validating the accuracy of data based on a pre-defined set of conditional logic. An example is to check an age to see if it falls in a range >= 18 and <= 65, then setting a variable to a value based on the results of that range check.

```
CREATE FUNCTION PrettyPhone
 (
      @phonein char(10)
 )
RETURNS char(15)
BEGIN
  RETURN '(' + SUBSTRING(@phonein,1,3) + ') ' +
SUBSTRING(@phonein,4,3) + '-' + SUBSTRING(@phonein,7,4)
END
GO
```

```
SELECT EmpID,
       EmpName,
       dbo.prettyphone(empphone) OfficePhoneNumber
FROM   Emp;
```

Stored Procedures

- SQL code that is compiled and stored as a database object.
- Can be used for a variety of database functions:
  - Change data: can include INSERT, DELETE, UPDATE statements.
  - Look at data: SELECT statements.
  - Accept parameters.
- Used to "bind together" related database statements so they can be executed together.
- Can be executed by a person, or by the computer.
  - Can be executed as a "job" at a given time of the day.
  - Can be executed by another program written in a different programming language.

Examples of use (stored procedures)

- Move purchase orders that have been fully received from a current purchase order table to a past purchase order table.
- Take detailed data (such as each row of an PurchaseOrderLine and related Receiver table) and create a summarized table that can be used for decision making.
- Search a table such as a list of vacation requests and send email about the status of requests that have been sitting in that table for more than 48 hours and needs to be processed by a person.

Common components of a procedural programming language

- Declare objects.
- Accept parameters.
- Do conditional logic (IF statements).
- Do loops.

Example of a stored procedure

```
CREATE PROCEDURE upSeeOrders AS
DECLARE @NameIn  VARCHAR(25)
SET @NameIn = ‘Johnson'

IF      (SELECT COUNT(*)
        FROM ord
        INNER JOIN emp
        ON ord.empid = emp.empid
        WHERE empname = @NameIn) > 0
--THEN
        SELECT *
        FROM ord
        INNER JOIN emp
        ON ord.empid = emp.empid
        where empname = @NameIn
ELSE
PRINT @NameIn + ' ' + 'has accepted no orders'
```

Triggers

- A specialized stored procedure that automatically executes when a database modification event occurs
  - Executes when an INSERT, UPDATE or DELETE event occurs.
  - Can be triggered prior to the event, or after the event.
- A trigger is attached to a single, specified table in the database.
- Unlike a stored procedure, a trigger cannot be initiated outside of the database modification event.
- Data validation.
  - CHECK constraint is limited.
  - A trigger can check multiple tables to validate data entry. For example, let's say that you want to validate the salary of a new employee. The salary should not be greater than the average salary for a similar job. The trigger code could look at the data in another table, calculate the average salary, and then compare it to the new salary entered. If the salary is not in alignment with the business rules, then the trigger would reject that transaction.
- Error report generation.
  - A trigger can validate data entry, and then write the rejected data to a new table.

Pop Quiz!

A stored procedure:

A. Is not available for use on SQL Server
B. Is written in ANSI-standard SQL
C. Is a database object that contains executable SQL code
D. Cannot include conditional logic or explicit program loops

Which of the following situations might best be handled with a SQL trigger?

A. Validating a salary when a new employee is input into an employee table.
B. Creating a standard format for a U.S. telephone number.
C. Calculating a BMI based on a person's weight and height.
D. Moving data to an archive table.

Which of the following situations might best be handled with a SQL function?

A. Validating a salary when a new employee is input into an employee table.
B. Deleting completing purchase orders from a current purchase order table.
C. Calculating a BMI based on a person's weight and height.
D. Moving data to an archive table.

What are the features of a DBMS that support partitioned applications?

- Discussed today
  - Stored program components: Functions, stored procedures, triggers
- Will discuss on Wednesday (04/23)
  - Concurrency control.
  - Database security.
  - Backup and recovery of defined transactions.

IS475/675 Agenda: April 23, 2025

- Finish discussing how database programming fits within a partitioned application environment.
- Describe features of a DBMS that support partitioned applications.
  - Concurrency control
  - Database security
  - Backup and recovery of pre-designated transactions

What is application partitioning?

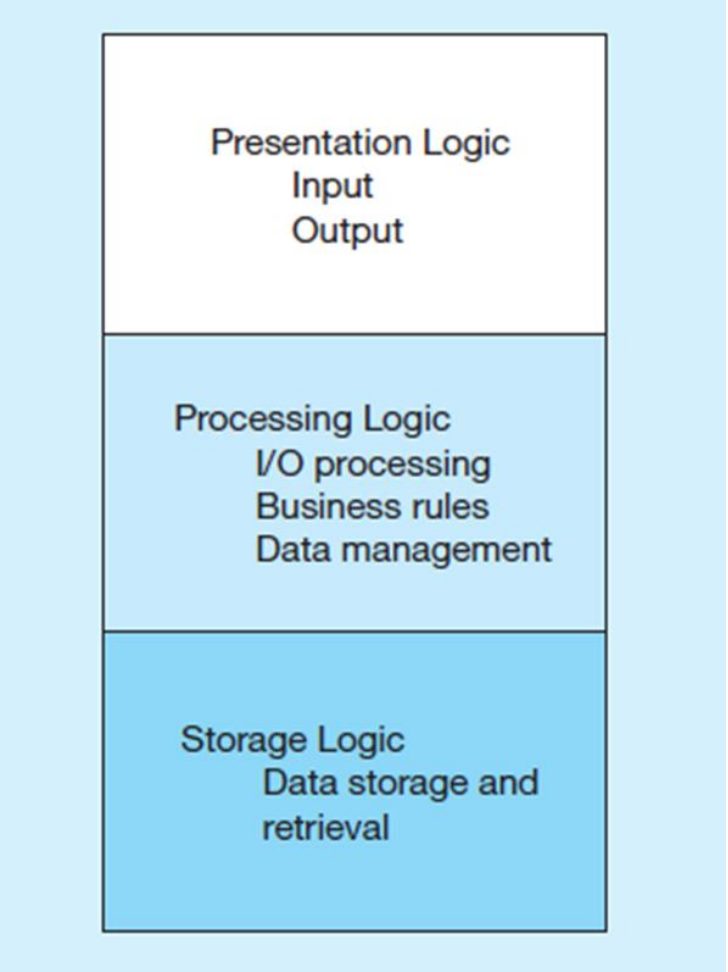
- Process of assigning portions of application code to client or server partitions.
- Described in Chapters 7 and 8 of the text "Modern Database Management"

Application Logic Components Depicted in Textbook

GUI interface (e.g. through a browser) that is accessed by the user of an application.

Procedures, functions, programs used to accomplish tasks within the application.

DBMS activities including inserting, updating, and deleting data within a database.



Four different types of procedural SQL programs

Type	Purpose	How it's usually executed	Where stored	Accepts parameters/ arguments?
Script	Execute multiple SQL statements	Within a client tool such as Management Studio	In a file on disk	No
Function	Automate a specific task	By an application or a SQL script	Database object	Yes
Stored Procedure	Execute multiple SQL statements in a pre-compiled format. Improve automation and efficiency.	By an application, by a person, or by a pre-specified time.	Database object	Yes
Trigger	Specialized stored procedure related to INSERT, UPDATE or DELETE.	Automatically by the database server when a specific transaction occurs.	Database object	No

Stored Procedures

- SQL code that is compiled and stored as a database object.
- Can be used for a variety of database functions:
  - Change data: can include INSERT, DELETE, UPDATE statements.
  - Look at data: SELECT statements.
  - Accept parameters.
- Used to "bind together" related database statements so they can be executed together.
- Can be executed by a person, or by the computer.
  - Can be executed as a "job" at a given time of the day.
  - Can be executed by another program written in a different programming language.

Examples of use (stored procedures)

- Move purchase orders that have been fully received from a current purchase order table to a past purchase order table.
- Take detailed data (such as each row of an PurchaseOrderLine and related Receiver table) and create a summarized table that can be used for decision making.
- Search a table such as a list of vacation requests and send email about the status of requests that have been sitting in that table for more than 48 hours and needs to be processed by a person.

Common components of a procedural programming language

- Declare objects.
- Accept parameters.
- Do conditional logic (IF statements).
- Do loops.

Example of a stored procedure

```
CREATE PROCEDURE upSeeOrders AS
DECLARE @NameIn  VARCHAR(25)
SET @NameIn = ‘Johnson'

IF      (SELECT COUNT(*)
        FROM ord
        INNER JOIN emp
        ON ord.empid = emp.empid
        WHERE empname = @NameIn) > 0
--THEN
        SELECT *
        FROM ord
        INNER JOIN emp
        ON ord.empid = emp.empid
        where empname = @NameIn
ELSE
PRINT @NameIn + ' ' + 'has accepted no orders'
```

Triggers

- A specialized stored procedure that automatically executes when a database modification event occurs
  - Executes when an INSERT, UPDATE or DELETE event occurs.
  - Can be triggered prior to the event, or after the event.
- A trigger is attached to a single, specified table in the database.
- Unlike a stored procedure, a trigger cannot be initiated outside of the database modification event.
- Data validation.
  - CHECK constraint is limited. A trigger is much less limited.
  - A trigger can check multiple tables to validate data entry. For example, let's say that you want to validate the salary of a new employee. The salary should not be greater than the average salary for a similar job. The trigger code could look at the data in another table, calculate the average salary, and then compare it to the new salary entered. If the salary is not in alignment with the business rules, then the trigger would reject that transaction.
- Error report generation.
  - A trigger can validate data entry, and then write the rejected data to a new table.

Pop Quiz!

A stored procedure:

A. Is not available for use on SQL Server
B. Is written in ANSI-standard SQL
C. Is a database object that contains executable SQL code
D. Cannot include conditional logic or explicit program loops

Which of the following situations might best be handled with a SQL trigger?

A. Validating a salary when a new employee is input into an employee table.
B. Creating a standard format for a U.S. telephone number.
C. Calculating a BMI based on a person's weight and height.
D. Moving data to an archive table.

Which of the following situations might best be handled with a SQL function?

A. Validating a salary when a new employee is input into an employee table.
B. Deleting completing purchase orders from a current purchase order table.
C. Calculating a BMI based on a person's weight and height.
D. Moving data to an archive table.

What are the features of a DBMS that support partitioned applications?

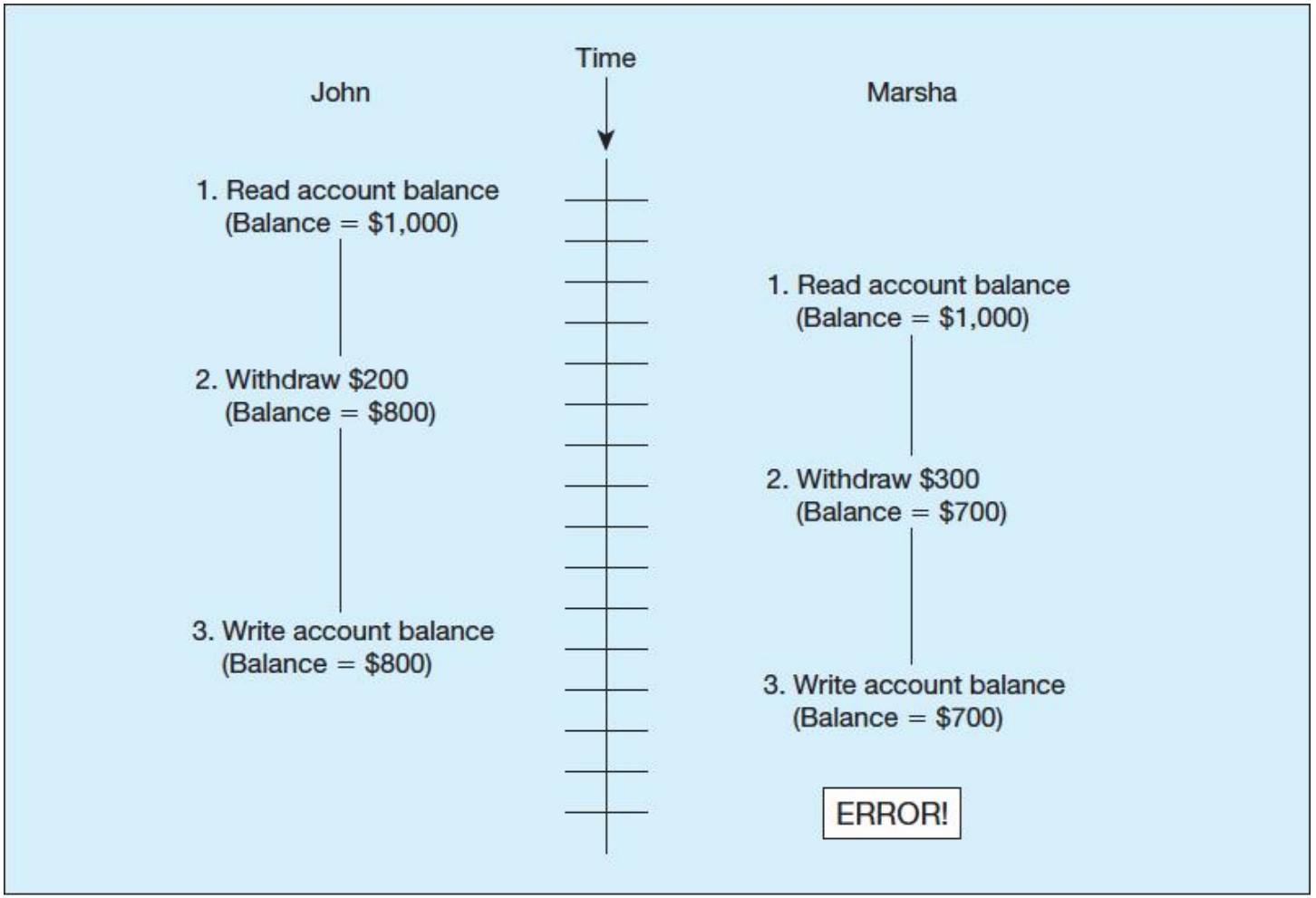
- Discussed so far today
  - Stored program components: Functions, stored procedures, triggers
- Now we need to discuss how a database management system (DBMS) supports partitioned applications.
  - Concurrency control.
  - Database security.
  - Backup and recovery of defined transactions.

Database Integrity: potential problems with shared databases

- Concurrency control is the process of managing concurrent operations against a database in order to maintain data integrity.

- Potential problems with shared databases are:
  - Lost Update.
  - Uncommitted Dependency. "Dirty Read"

Book Lost Update Example



Uncommitted dependency or "dirty read"

John	Time	Marsha
Read account balance (balance = $1,000)	T1	
Withdraw $200 (balance = $800)	T2	
Write account balance to memory (balance = $800)	T3	
[Remembers that Marsha needs $1,000 today]	T4	
	T5	Read account balance (balance = $800)
Rollback (transaction wasn't fully completed yet)	T6	
	T7	[Marsha is ticked off when she sees only $800]

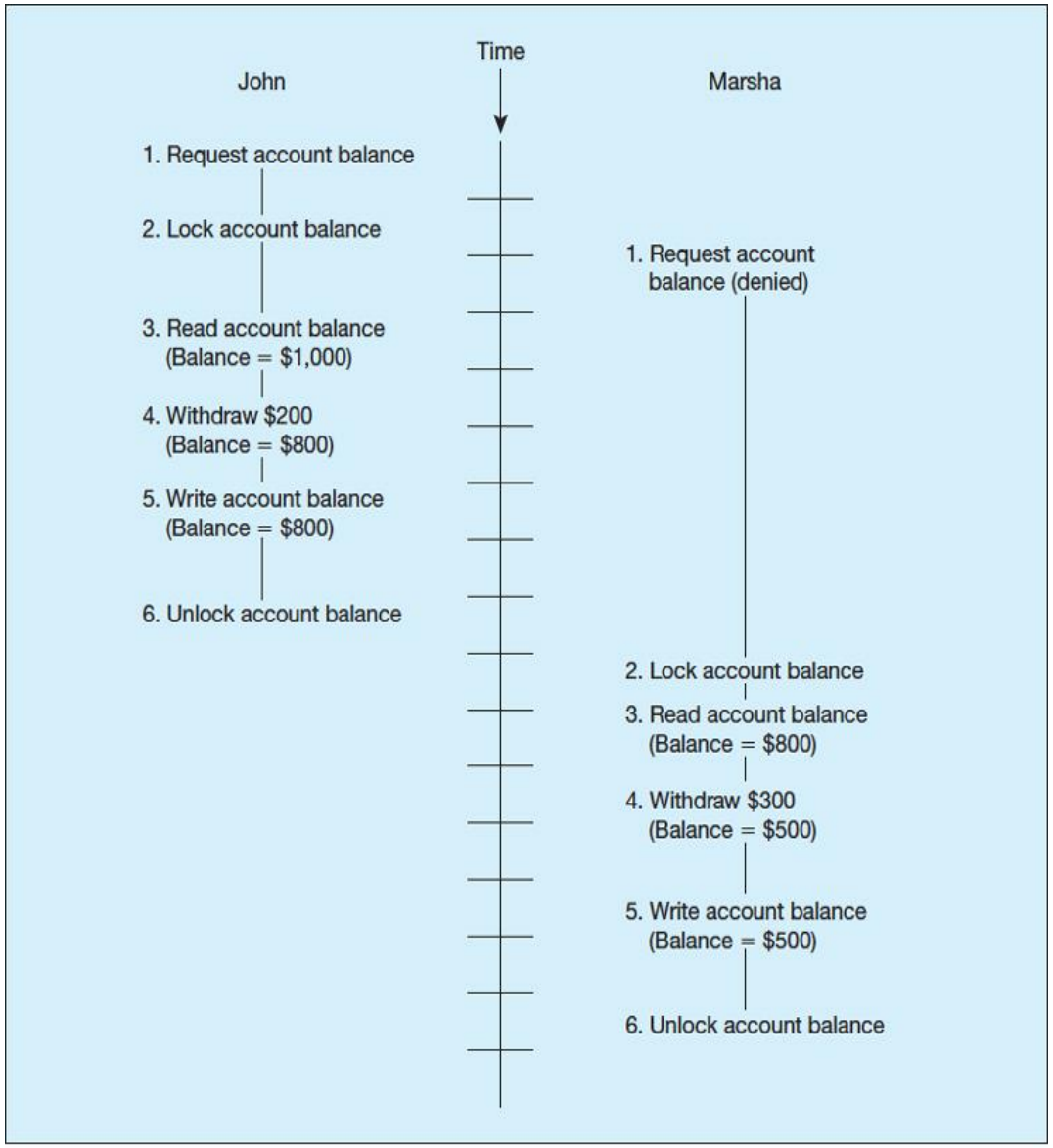
Methods of concurrency control

- **Pessimistic approach:**
  - Assumes that every transaction could potentially be in conflict so each transaction should be controlled
- **Optimistic approach:**
  - Assumes that most transactions will not be in conflict so each transaction should be checked only when something is written to disk.
- **Both approaches need a Scheduler.** The scheduler establishes the order in which the operations within concurrent transactions are executed.
  - The scheduler interleaves the execution of database operations to ensure serializability.
  - Think of it as based on a time-stamp.
  - Some schedulers have the ability to analyze transaction content.

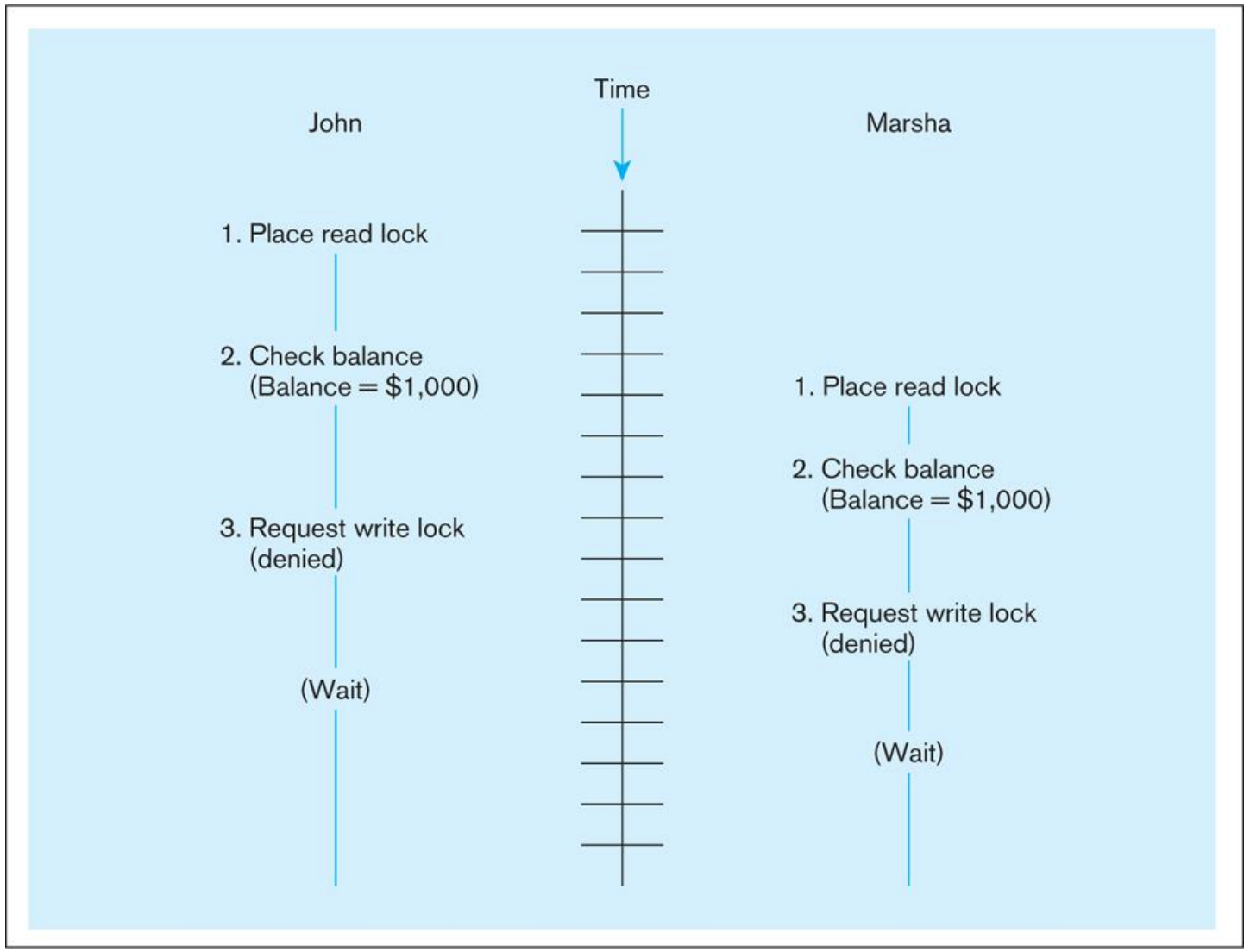
The pessimists use locking

- Locking: Fundamental tool of pessimistic concurrency control. A lock guarantees exclusive use of a data item to a current transaction.
- Locking. Locking can be performed programmatically or left to the DBMS.
  - Granularity of locking depends on the DBMS.
    - Database level.
    - Table level.
    - Page level.
    - Row level.
    - Column level.
- Obtain read lock before accessing an item.
- Wait if a conflicting lock is held.
  - Shared lock: conflicts with exclusive locks
  - Exclusive lock: conflicts with all other kinds of locks
- Concurrency control manager maintains the lock table

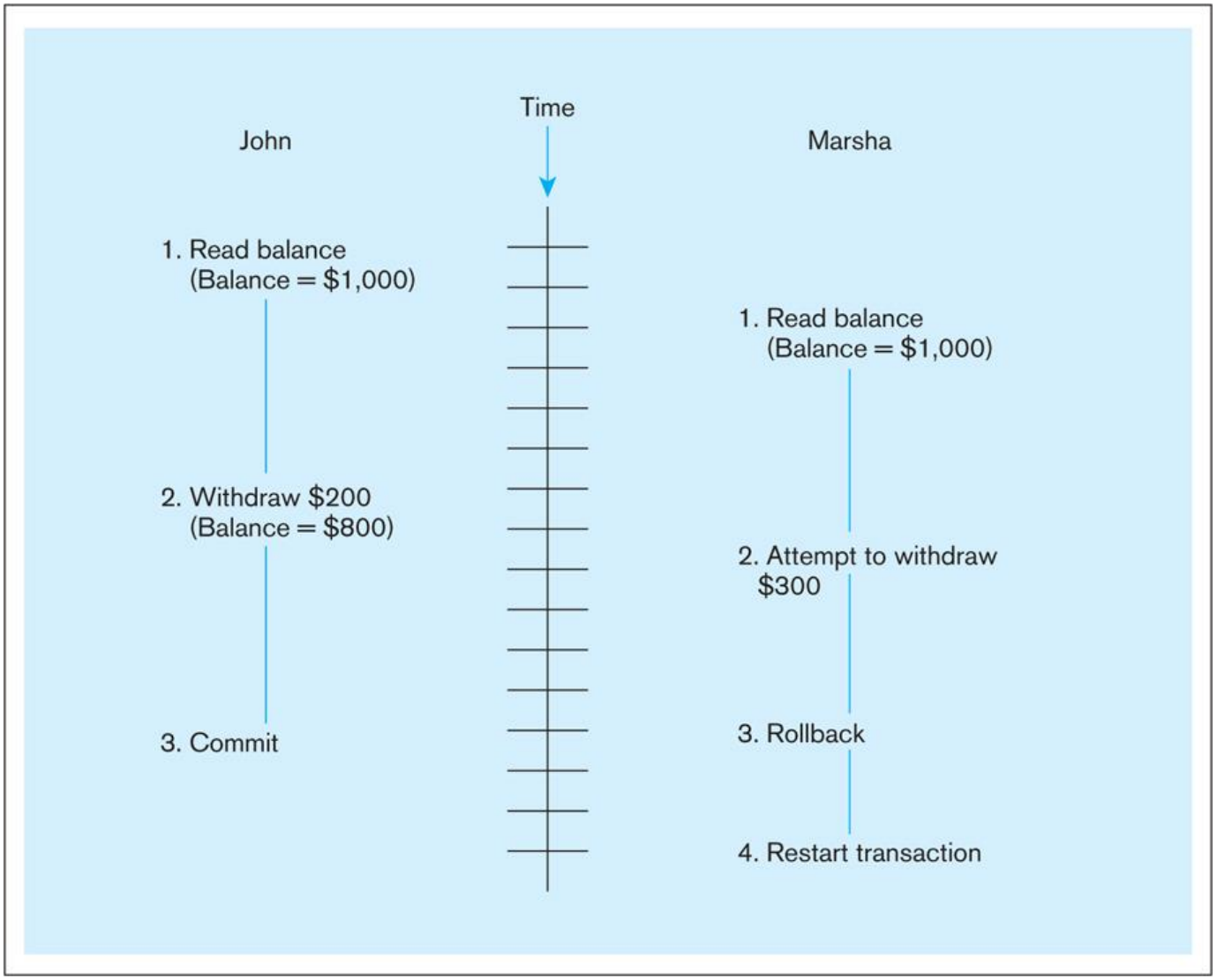
Lock it up!



Problem that can occur: Both transactions get read lock and create deadlock when requesting write lock



The Optimists use versioning



Versioning

- Assumes that most transactions read, rather than write, data.
- Each transaction uses the view of the database that is current when it starts.
- When a transaction modifies a record, the DBMS create a new version rather than rewriting the old record.
- If there is a conflict, the DBMS stores the first changed version, and the second transaction is cancelled.
- The second transaction takes the new version of the data and uses that for processing.

Pop Quiz!

The actions taken by a database management system to ensure data integrity is maintained during **multiple transactions with a shared database** are called:

a. transaction logging.
b. transaction timestamping
c. transaction authorization.
d. multiple user management.
e. concurrency control.

Consider pessimistic concurrency control. A lock on a larger data element, such as a table, versus a smaller data element, such as a row:

a. Increases waiting by other transactions.
b. Decreases waiting by other transactions.
c. Increases system overhead (i.e. use of main memory, processor and disk).
d. Decreases both system overhead and waiting by other transactions.

Database security

- Database Security: Protection of the data against accidental or intentional loss, destruction, or misuse. Threats to database security include the list to the right.

- Accidental losses attributable to:
  - Human error.
  - Software failure.
  - Hardware failure.
- Theft and fraud.
- Improper data access:
  - Loss of privacy (personal data).
  - Loss of confidentiality (corporate data).
- Loss of data integrity.
- Loss of availability.

Top **Database** Security Threats

- **Privilege abuse:** excessive access privileges, using privileges to create unauthorized data sets, enhancing privilege access from user to administrator.
- **SQL Injection:** adding SQL code via web forms (or other vulnerable input channels) to execute fraudulent commands.
- **Lack of encryption** of identifying data.
- Weak authentication.
- Backup data availability.
- Weak audit trail.

SQL Injection

```
SELECT email, passwd, login_id, full_name
FROM     members
WHERE email = '$email'
```

So, now imagine that someone enters something like this for the email address:
Anything' or 'x' = 'x

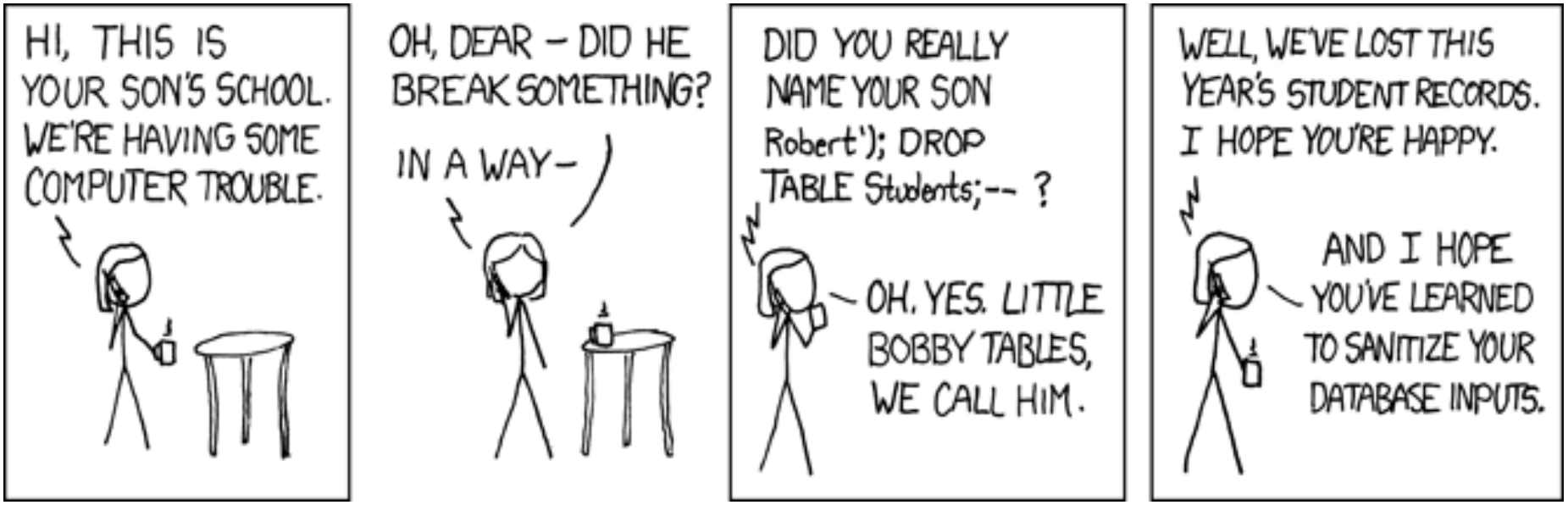
```
SELECT email, passwd, login_id, full_name
FROM     members
WHERE email = 'Anything' or 'x' = 'x'
```

Two very depressing websites…

http://www.unixwiz.net/techtips/sql-injection.html

http://www.w3schools.com/sql/sql_injection.asp

And a cute comic…



DBMS security features

- Views (frequently referred to as subschemas).
- Integrity controls.
- Authorization rules.
  - Controls incorporated in the DBMS.
  - Restrict access to specific data.
  - Restrict actions that can be taken.
- User-defined procedures.
  - Trigger an authorization procedure which asks additional identification questions.
  - Written in a standard programming language or proprietary language.
- Stored procedures for all data input/access to the database.
- Encryption.
- Authentication schemes.
  - Biometric devices.

SQL statements used for security

SQL Statement	Action
CREATE USER	Allows the DBA to create a new user.
GRANT	Allows the user to give other users privileges to access the user's objects.
CREATE ROLE	Allows the DBA to create a collection of privileges that can be assigned as a group.
ALTER USER	Allows users to change their passwords. Can also be used to change other attributes of a user.
REVOKE	Removes privileges on an object from a user, users, or role.

Pop Quiz!

Which of the following statements regarding database security is TRUE?

a. A database management system does not have any methods of ensuring appropriate access to data in the database.
b. A view is not used to control access to tables in a database.
c. All data stored in a database should be encrypted.
d. Some database management systems provide a way to encrypt data.

Which of the following database security measures can help protect against SQL injection attacks?

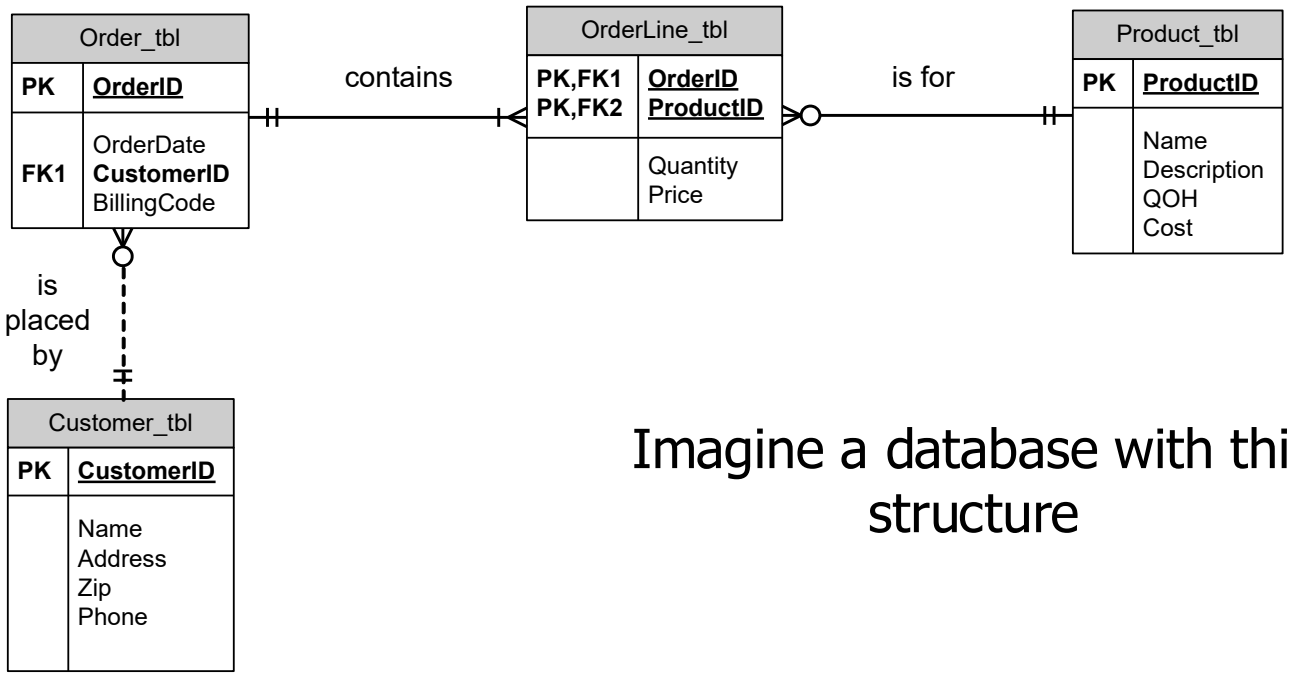
a. Using authorization rules to prevent access to a database.
b. Using database constraints such as referential integrity.
c. Using stored procedures to check data input fields to see if they contain SQL code.
d. Using views to prevent programmer access directly to database tables.

Database backup and recovery

- What is backup and recovery?
  - **Backup:** A method of storing data from a database in a format that can be used to rebuild the database very quickly if necessary to rebuild.
  - **Recovery:** Mechanisms for restoring a database very quickly and accurately after loss or damage.
- Why have backup and recovery?
  - Human error.
  - Hardware failure.
  - Incorrect or invalid data.
  - Program errors.
  - Viruses.
  - Natural catastrophes.

Backup and recovery are based on transactions

- A transaction is one or more database actions (SQL statements) that are treated as a single unit of work.
  - If the transaction is successful, then the transaction is **committed**.
  - If the transaction is not successful, then the transaction is **rolled back** or **aborted**.



Imagine a database with this structure

Accepting an order for a product

```
INSERT INTO order_tbl VALUES
    (123, '26-apr-2025', 765, 'net30');
```

```
INSERT INTO orderline_tbl VALUES
    (123, 6812, 10, 34.99);
```

```
UPDATE    product_tbl
SET       qoh = qoh - 10
WHERE     prod_no = 6812;
```

A transaction is:

Atomic: The transaction cannot be sub-divided; it must all be completed and saved to disk for it to be valid.

Consistent: The rules and constraints of the database are consistent for any data written to disk.

Isolated: Transactions act like they are running separately from any other transactions.

Durable: Changes to disk are permanent and require another transaction to occur for any modifications to happen.

DBMS's have methods to control transactions

- Databases that support transactions provide specific commands for starting, committing, and rolling back transactions.
  - Begin transaction.
  - End transaction.
  - Commit.
  - Rollback.
  - Autocommit.

Database backup is conducted in 3 processes

Backup: A DBMS software utility provides a way to do a complete, full or incremental backup of the database in a consistent state.

- Complete: entire database copied at pre-specified times. Not real-time.
- Incremental: rows that have changed since the last full backup.
- Mirror image: A copy of the database stored in a separate location on a separate device that is updated in real-time.

Journalize: A DBMS software utility provides an audit trail of changes to the database.

- Transaction log: contains all data used to process changes against the database. Maintained real-time.
- Database change log: contains a before-image and an after-image of each row modified by a database transaction. Maintained real-time.

Checkpoint: A DBMS software utility that periodically suspends all transaction processing and synchronizes files within the database.

- Some databases, such as Oracle, do not actually halt processing. They simply write checkpoint information to physical files.
- The purpose of a checkpoint is to minimize the amount of time it takes to restore a system.

Recovery methods

- A DBMS has a utility to recover the database. Usually referred to as the Recovery Manager.
- The method of recovery depends on the type of failure.
- Recovery Manager usually has the following options:
  - **Switch:** Switches to a replica of the database on a different storage device.
    - Requires that a **mirror image** of the database is stored.
    - Can be expensive.
    - Assumption is a storage failure, not a failure in transaction integrity, occurred.
  - **Restore/Rerun:** Reprocesses the transactions for a given time period against a correct version of the database.
    - Assumption is that a failure in transaction integrity has occurred.
    - Can be very time-consuming.

Issues in database backup and recovery

- Cost.
  - Media.
  - Computer overhead (processor, memory, disk) to create journalizing files, control files, checkpoint files, etc.
  - Personnel to supervise and tune.
- Time.
  - Can result in regularly scheduled downtime.
  - Can make the system slower.

Pop Quiz!

A discrete unit of work that must be processed completely or not at all when using a database management system is called a:

a. before image.
b. journalizing facility.
c. ERD.
d. transaction.
e. subschema

A large, global organization such as FedEx, that requires 24 hours a day/ 7 days a week access to its database, would probably choose which of the following recovery methods?

a. rollback.
b. restore/rerun.
c. switch.
d. checkpoint utility.
e. transaction authorization.