Let pureQuery improve the quality of service and reduce costs for WebSphere and DB2 applications

Stephen Brodsky
Holly Hayes

February 2009

Frequently Cited Concerns

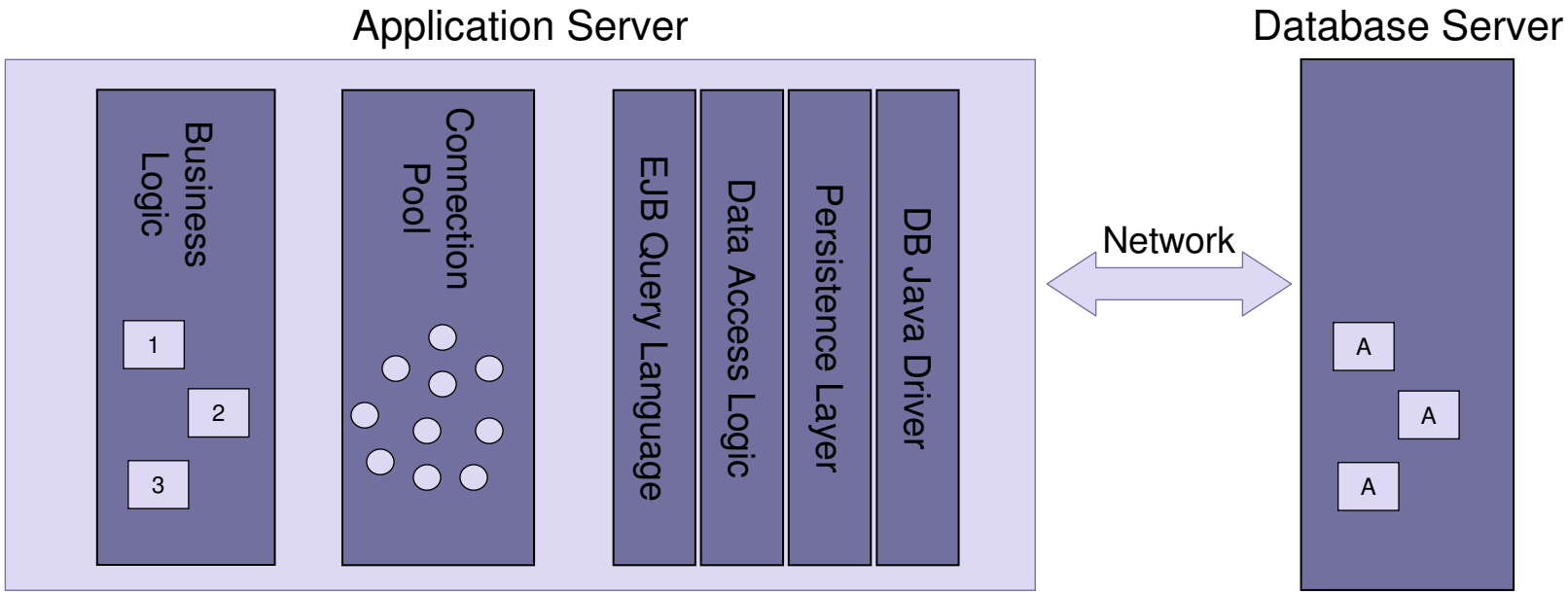
I have more and more Java workload coming onto my mainframe driving up costs, but the budget is not keeping pace.

I don't even want to allow framework-generated SQL onto the mainframe. If I can't see it, I don't know how it will impact me.

Java performance problems are a real pain to resolve because I can't even tell what application issued the SQL.

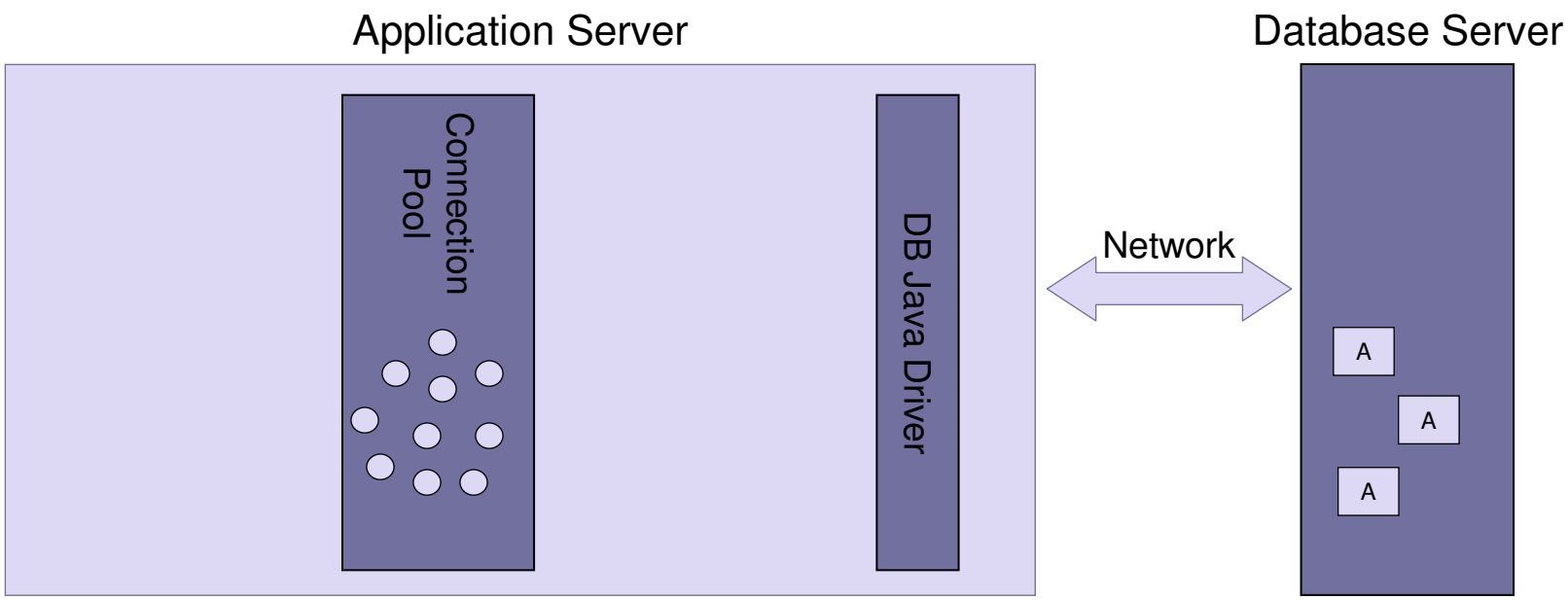
Contemporary Application Stack Challenges

- **Simplify development, but …**
  - Challenge problem resolution
  - Impact performance
  - Obscure impact analysis
  - Impede capacity planning



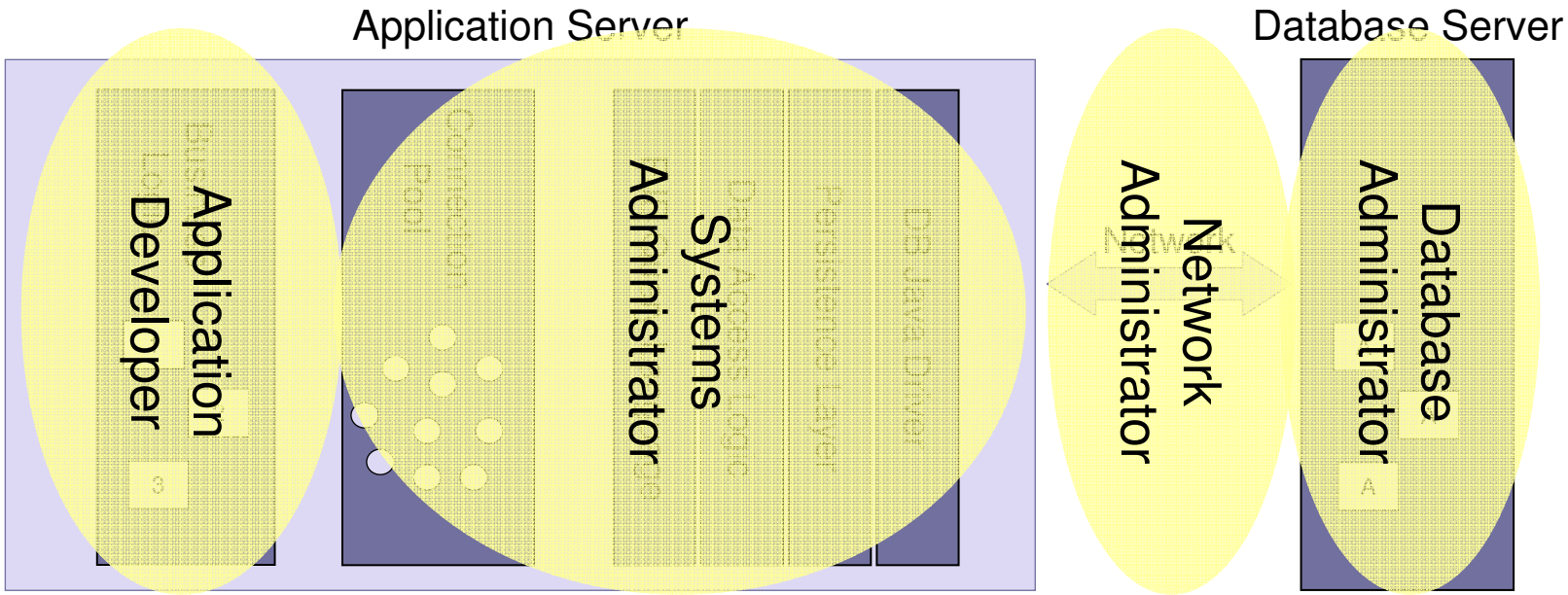
DBA Perspective
Layers obscure linkages

- **What is visible to the DBA?**
  - SQL statement
  - Database resource consumption
  - IP address of application server
  - Connection pooling userid
  - Application is running JDBC or CLI
- **What is not known by the DBA?**
  - Which application is running?
  - Which developer wrote the application?
  - What other SQL does this application issue?
  - When was the application last changed?
  - How has CPU changed over time?.....



Problem Resolution, Impact Analysis, and Capacity Planning

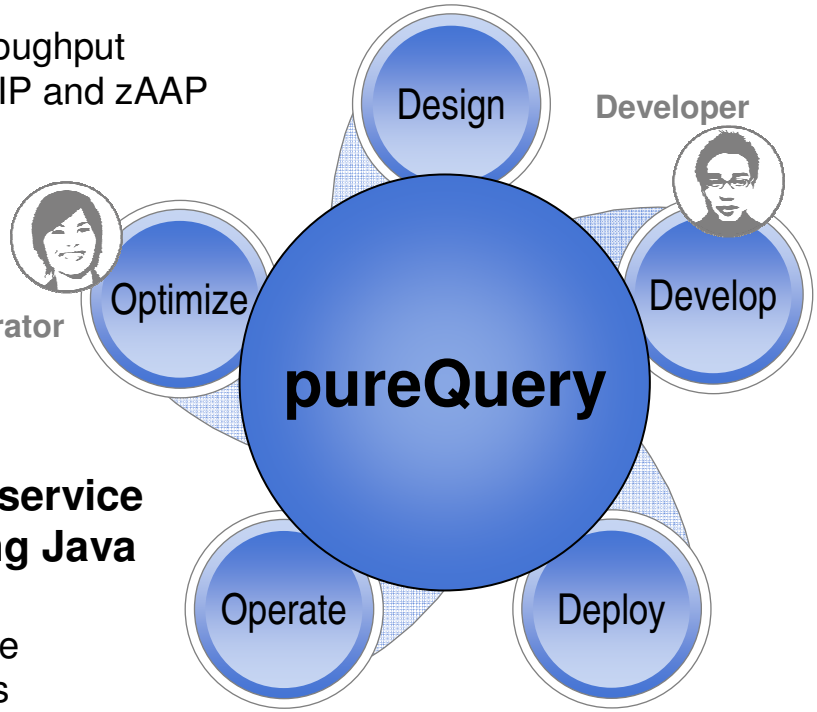
- **Time-consuming, staff-intensive processes**
  - No holistic view
  - No resource consumption profile
  - No correlation of SQL to source application
  - Problem isolation requires cross-role collaboration



Build Better Applications, Faster with pureQuery

Reduce costs
- Increase system throughput
- Move workload to zIIP and zAAP

Reduce time to market for new Java applications
- Bridge Java and data
- Balance productivity and control
- Tune before deployment
- Enhance developer and DBA collaboration



Improve quality of service for new and existing Java applications
- Improve performance
- Lock in access plans
- Speed up problem resolution
- Improve impact analysis and capacity planning

Enhance security
- Limit user access
- Minimize SQL injection risk
- Improve audit readiness

Introducing pureQuery

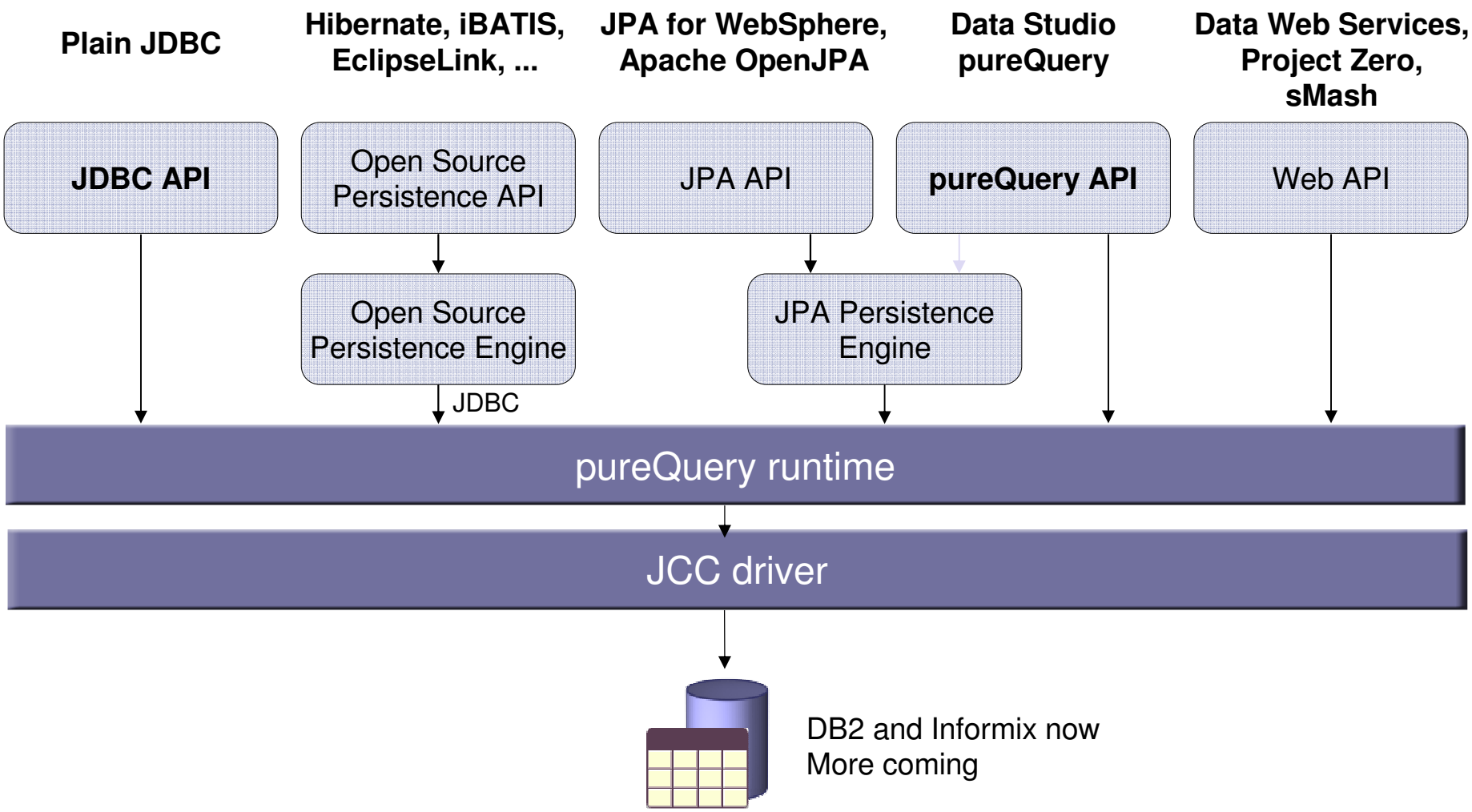
A high-performance, data access platform to simplify developing, managing, securing, and optimizing data access for new and existing applications.

pureQuery Components:

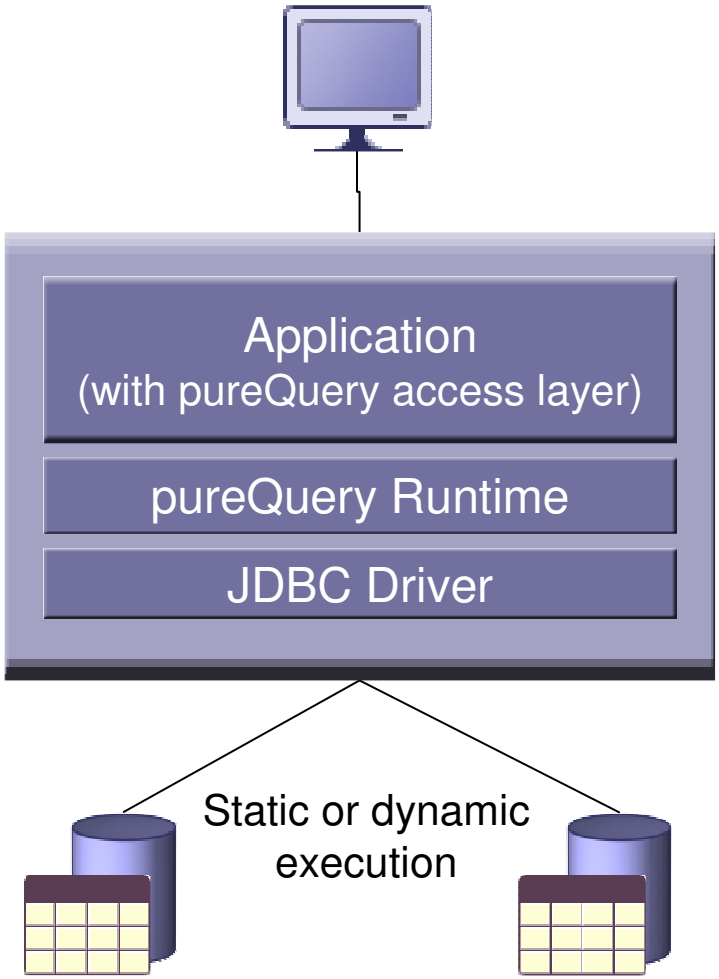
- **Data Studio Developer**
  - Integrated development environment with Java and SQL support
  - Improve problem isolation and impact analysis
- **Simple and intuitive API**
  - Enables SQL access to databases or in-memory Java objects
  - Facilitates best practices
- **Data Studio pureQuery Runtime**
  - Flexible static SQL deployment for DB2

Java Database Access and pureQuery

Many on-ramps for new and existing applications



Deploying with pureQuery Runtime



Application tier:

- z/OS, Linux, UNIX, Windows

Database tier:

- DB2 for z/OS
- DB2 for i
- DB2 for Linux, UNIX, and Windows
- Informix Dynamic Server

pureQuery Improves Performance, Security, and Manageability for DB2
…Without Changing a Line of Code

Three steps

1. **Capture the SQL**
   - Use pureQuery API, generate from WebSphere JPA, or capture while executing
   - Use with custom-developed, framework-based, or packaged applications
2. **Bind SQL to DB2**
   - Use tooling in Data Studio Developer, WAS console or command line
3. **Choose execution mode**
   - Dynamic or static
   - Choose at deployment time instead of development time

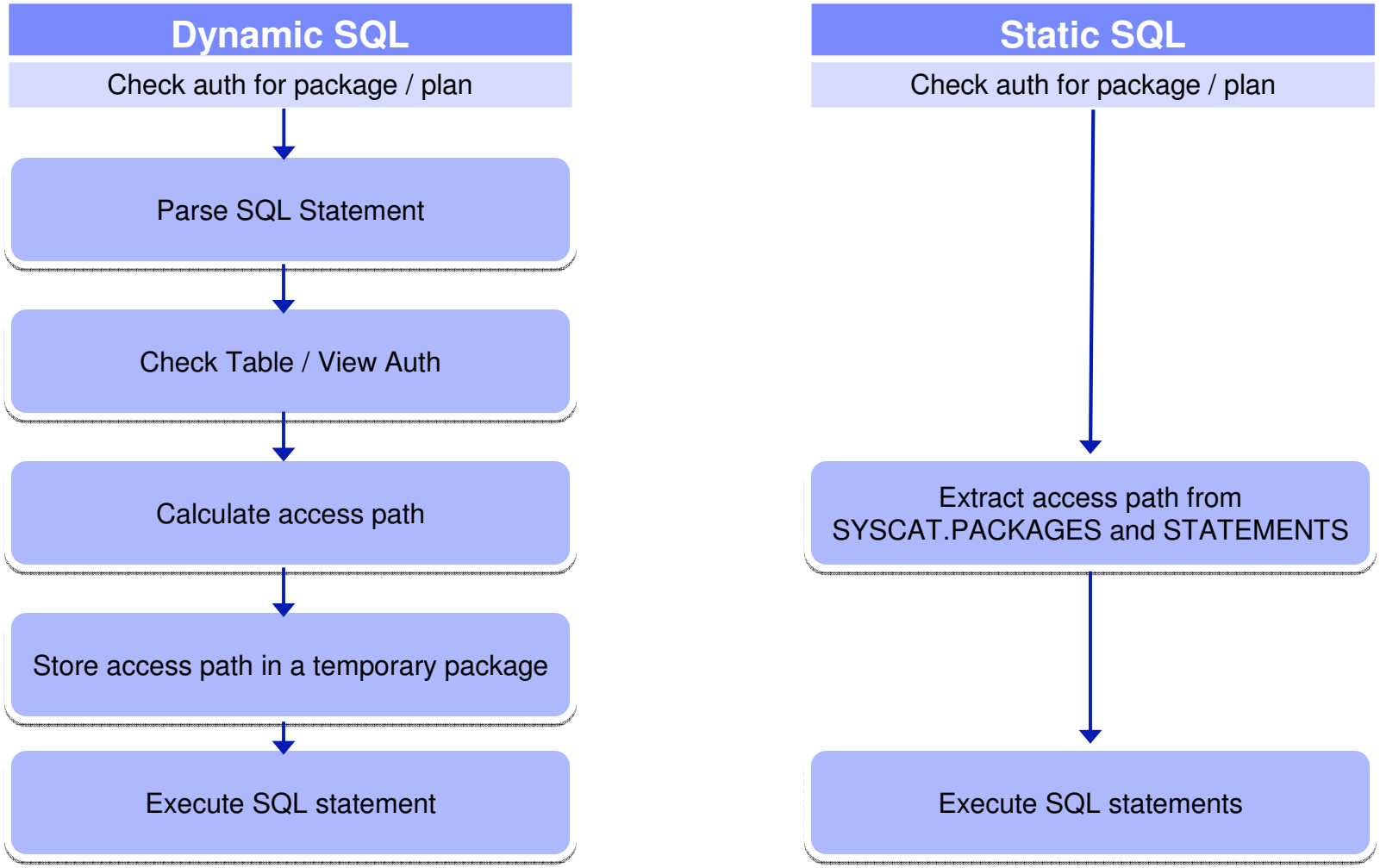
Static SQL value

- **Make response time predictable**
  - Lock in the SQL access path pre-execution
- **Limit user access and reduce SQL injection**
  - Grant execute privileges on the query packages rather than access privileges on the table
- **Accelerate problem resolution**
  - Trace SQL execution to a specific package and the originating source
- **Improve impact analysis and capacity planning**
  - Visualize application SQL and correlation metadata
- **Increase system capacity**
  - Drive down DB cycles

"The ability to use static SQL with pureQuery is huge. Recently, I worked with a client who could **reduce CPU usage by 7 percent** thanks to this one feature."

— David Beulke, Pragmatic Solutions Inc.

Dynamic vs. Static Execution

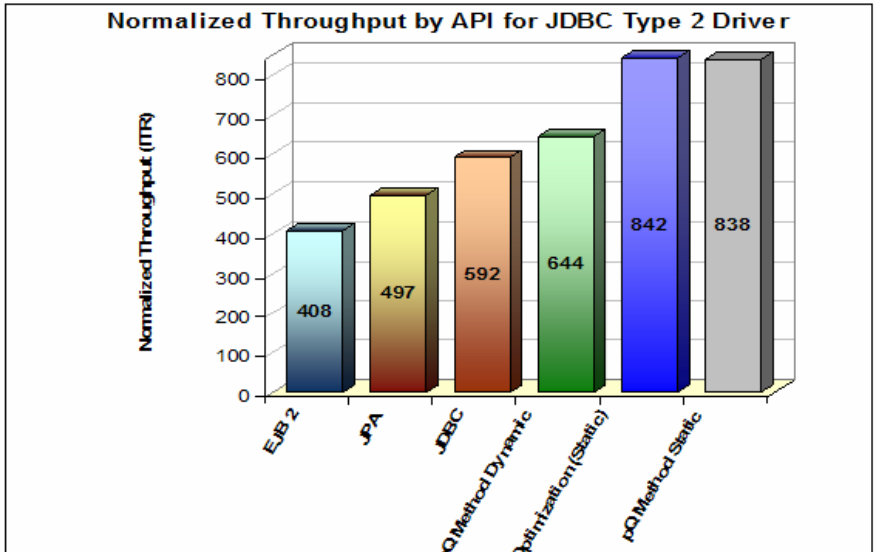


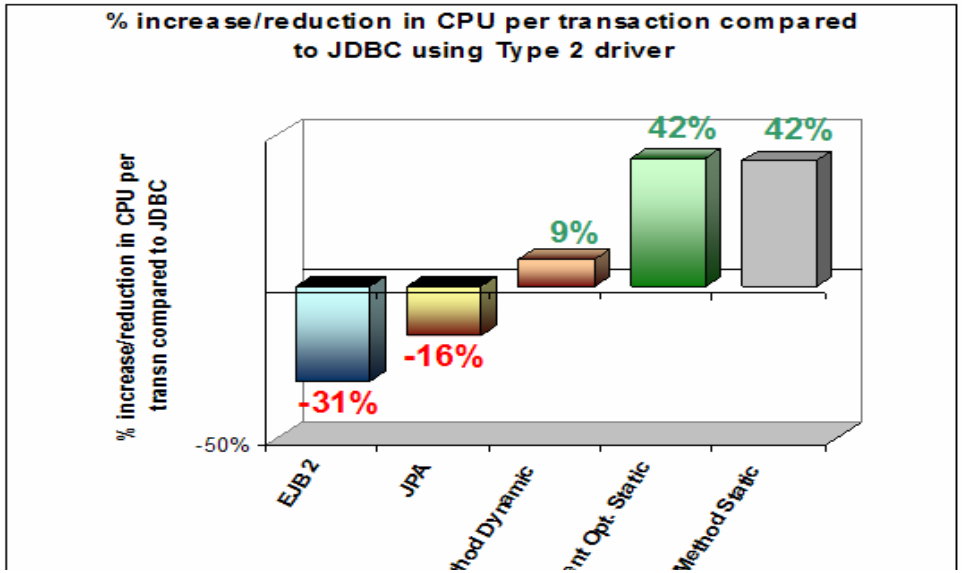
Static Execution Advantages

Feature	Dynamic SQL (pureQuery, JDBC)	Static SQL (pureQuery, SQLJ)
Performance	Can approach static SQL performance with help from dynamic SQL cache. Cache misses are costly	All SQL parsing, catalog access, done at BIND time. Fully optimized during execution.
Access path reliability	Unpredictable – Any prepare can get a new access path as statistics or host variables change	Guaranteed – locked in at BIND time All SQL available ahead of time for analysis by EXPLAIN.
Authorization	Privileges handled at object level. All users or groups must have direct table privileges – Security exposure, and administrative burden	Privileges are package based. Only administrator needs table access. Users/Groups have execute authority. Prevent non-authorized SQL execution.
Monitoring, Problem determination	Database View is of the JDBC or CLI package – No easy distinction of where any SQL statement came from.	Package View of applications makes it simple to track back to the SQL statement location in the application
Capacity planning, Forecasting	Difficult to summarize performance data at program level.	Package Level Accounting gives program view of workload to aid accurate forecasting.
Tracking dependent objects	No record of which objects are referenced by a compiled SQL statement	Object dependencies registered in database catalog

Improving Throughput with pureQuery, a z/OS Example

- **In-house testing shows over 40%reduction in CPU costs over dynamic JDBC**
  - z/OS pureQuery Benchmark: IBM Data Studio pureQuery Runtime for z/OS Performance http://www.ibmdatabasemag.com/story/showArticle.jhtml?articleID=208802229
  - IRWW – an OLTP workload, cache hit ratio between 70 and 85%, Type 2 Driver





Any performance data contained in this document were determined in various controlled laboratory environments and are for reference purposes only. Customers should not adapt these performance numbers to their own environments as system performance standards. The results that may be obtained in other operating environments may vary significantly. Users of this document should verify the applicable data for their specific environment.

Unique Package Names Improves PD

- Most dynamic Java applications use packages SYSLNx00 making it hard to identify specific programs
- Unique package names link SQL to Java Beans, similar to CICS transaction names to programs.

Static *pureQuery* Java SQL

Dynamic Java SQL

```
                 ZALLU    VTM     02       V410./C DB1S 09/12/08 11:29:22   2
> Help PF1      Back PF3       Up PF7      Down PF8     Sort PF10    Zoom PF11
> T.A                          OMEGAVIEW PA2
>         THREAD ACTIVITY:  Enter a selection letter on the top line.

> *-ALL         B-TSO      C-CICS      D-IMS     E-BACKGROUND   F-DIST ALLIED
> G-DIST DBAC   H-UTIL     I-INACT     J-FILTER  K-FUNCTIONS    L-STORED PROC
> M-TRIGGERS    N-SYSPLEX  O-ENCLAVES  P-WORKSTA
==============================================================================
>                       ALL THREADS CONNECTED TO DB2
PTHDA                                                                FLTR ON
+               *
+ Elapsed       Package   CPU    Status       GetPg  Update Commit CORRID
+ ---------     --------  -----  ------------ ------ ------ ------ ------------
+ 00:00:13.6    PAW_OR_O  00.0%  IN-DB2           25      0      0 db2jcc_appli
+ 00:02:27.3    SYSLN200  00.0%  IN-DB2          897      0      0 db2jcc_appli
+ 00:02:52.3    SYSLN200  00.0%  IN-DB2         1025      0      0 db2jcc_appli
+ 00:02:[illegible].8    SYSLN200  00.0%  IN-DB2         1324      0      0 db2jcc_appli
+ 00:02:32.7    SYSLN200  00.0%  IN-DB2          961      0      0 db2jcc_appli
+ 00:02:59.2    SYSLN200  00.0%  IN-DB2         1046      0      0 db2jcc_appli
==============================================================================
```

Reduce Costs with zIIP and zAAP

- **COBOL or SQL/PL stored procedures often instead of executing SQL directly**
  - Developers need not concern themselves with writing efficient SQL
  - DBAs retain better control over SQL including static execution
  - If not written in DB2 9 for z/OS's Native SQL/PL, the stored procedure must use general purpose processors

- **pureQuery introduces alternatives**
  - Create Java stored procedures to run on zAAP using the pureQuery runtime
    - DBAs retain control, Data Studio helps with development, pureQuery executes statically
  - Execute SQL directly from Java application or method to run on zIIP
    - Developers use Data Studio Developer to generate access layer with pureQuery, content assist helps with best practices and SQL validation, packages SQL for easy collaboration with DBA, pureQuery executes statically

IBM Data Studio Developer

An integrated database development environment
that speeds application design, development, and deployment
while increasing data access performance and manageability.

- **Enhance developer productivity**
  - Drag and drop creation of Web services
  - Provide a seamless SQL/Java experience
  - Generate a data access layer using Java objects, JSON, or, XML
  - Enhance problem isolation and impact analysis, even when using frameworks that generate the SQL
- **Provide expert-equivalent performance**
  - Facilitate use of JDBC and SQL data access best practices
  - Improve DB2 performance, predictability, and manageability by enabling transparent activation of static SQL (i.e. no change to the application) for Java and .NET
  - Facilitate DBA collaboration and optimization
- **Enhance security**
  - Eliminate SQL injection risk
  - Minimize access privileges

UNIVAR

"IBM Data Studio enables us to bridge the gap between object-oriented design and relational database technology. As a result, we can speed the development of high quality applications and improve developer productivity by between 25 and 50 percent"

pureQuery Balances Productivity and Control

JDBC / SQLJ — Code all your SQL

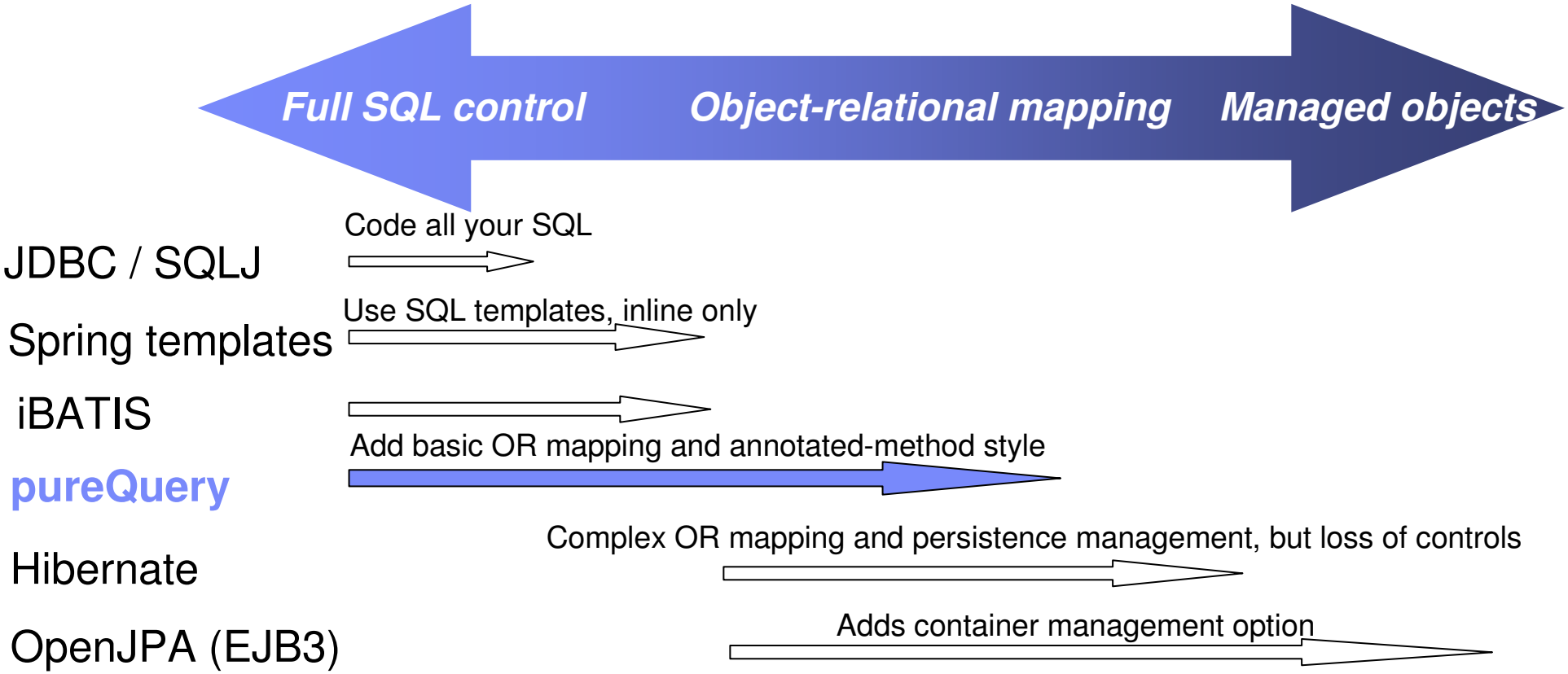
Spring templates — Use SQL templates, inline only

iBATIS

Hibernate — Complex OR mapping and persistence management, but loss of controls

OpenJPA (EJB3) — Adds container management option

pureQuery Balances Productivity and Control



pureQuery Facilitates Best Practices

- **Supports both inline SQL and Java annotations (method)**
- **Intuitive interfaces for common data retrieval and manipulation scenarios hides JDCB complexity**
  - Query First
  - Homogeneous Batch
- **Reduce network trips to the database**
  - Query Over Java Collections
  - Heterogeneous Batch
- **Use custom result handlers to map results to POJO's, XML, JSON, …**

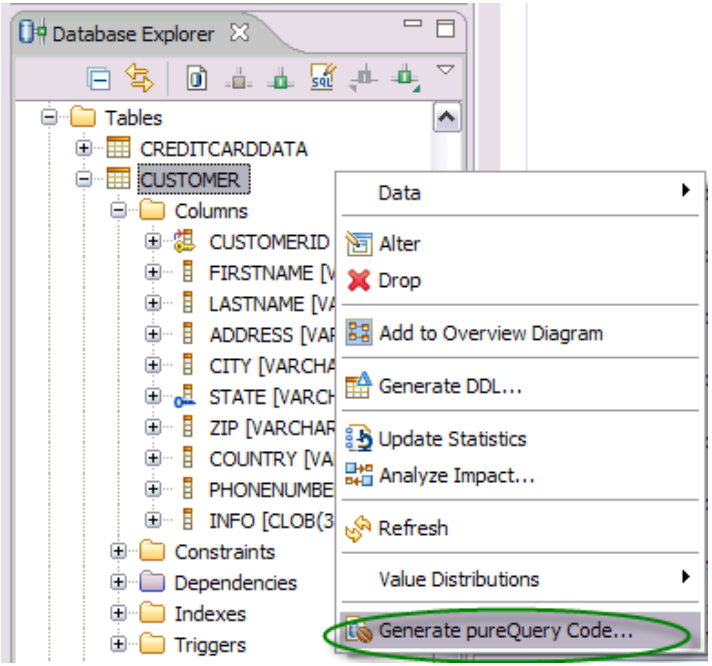
Write high performance Java data access applications, Part 3: Data Studio pureQuery API best practices

—Vitor Rodrigues

http://www.ibm.com/developerworks/db2/library/techarticle/dm-808rodrigues/?S_TACT=105AGX01&S_CMP=LP

Java Data Access in 5 simple steps

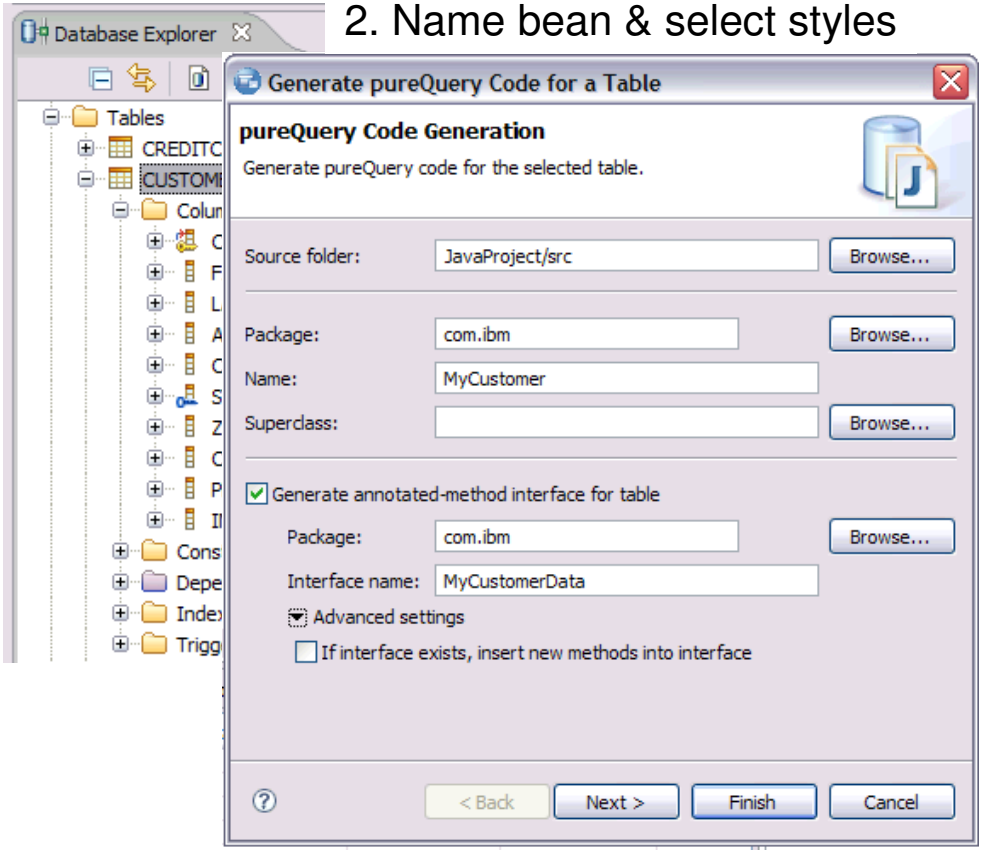
1. Select table



Java Data Access in 5 simple steps

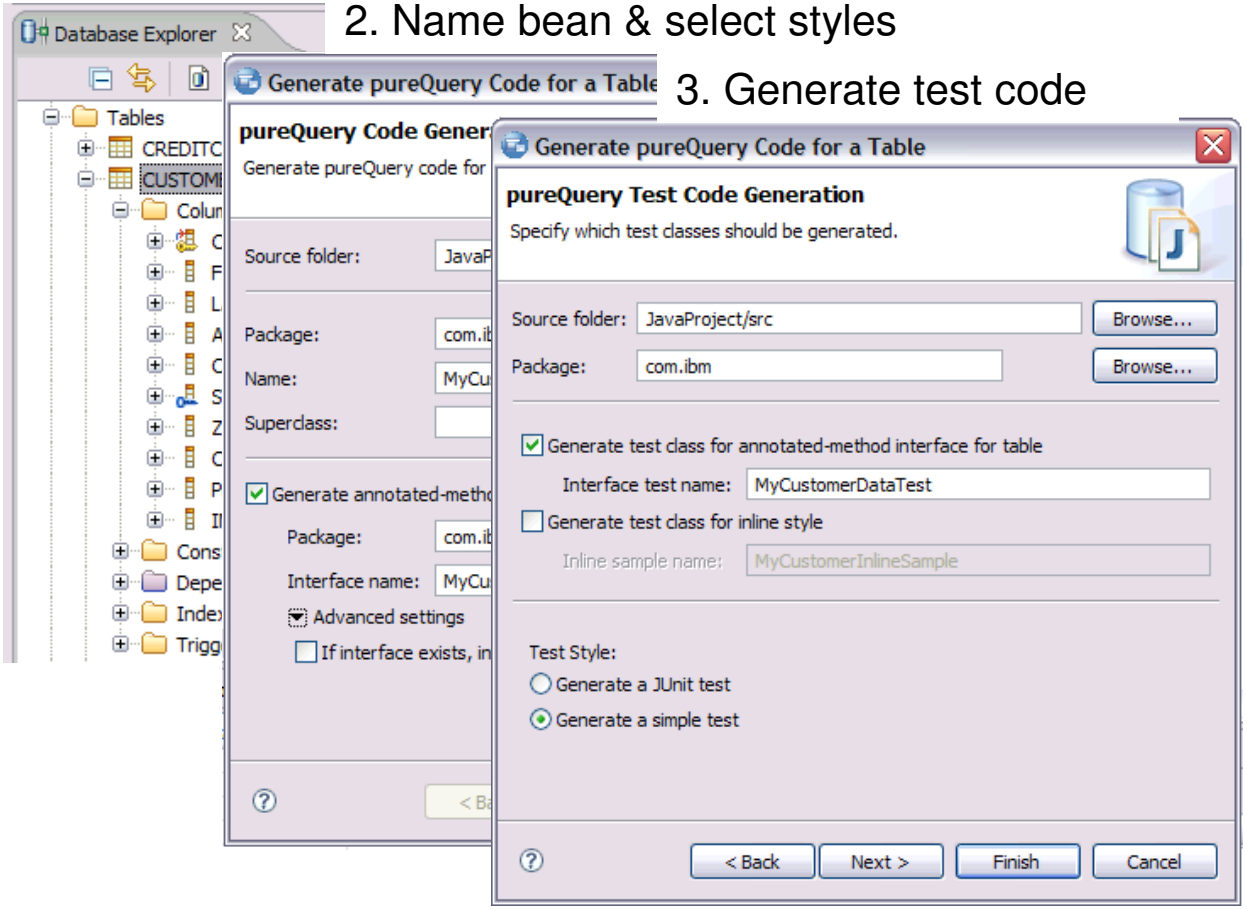
1. Select table

2. Name bean & select styles



Java Data Access in 5 simple steps

1. Select table
2. Name bean & select styles
3. Generate test code



Java Data Access in 5 simple steps

1. Select tab
2. Name bean & select styles
3. Generate test code
4. Map table to bean

Database Explorer

Tables

Generate pureQuery Code for a Table

pureQuery Code Gener

Generate pureQuery code for

Source folder: JavaP

Package: com.ib

Name: MyCu

Superclass:

Generate annotated-metho

Package: com.ib

Interface name: MyCu

Advanced settings

If interface exists, in

Generate pureQuery Code for a

pureQuery Test Code G

Specify which test classes sho

Source folder: JavaProject/s

Package: com.ibm

Generate test class for ar

Interface test name:

Generate test class for inl

Inline sample name:

Test Style:

Generate a JUnit test

Generate a simple test

Generate pureQuery Code for a Table

Bean Fields

Specify how to define the bean fields.

Select the scope of the bean fields:

Public fields with no accessor or mutator methods

Protected fields with public accessor and mutator methods

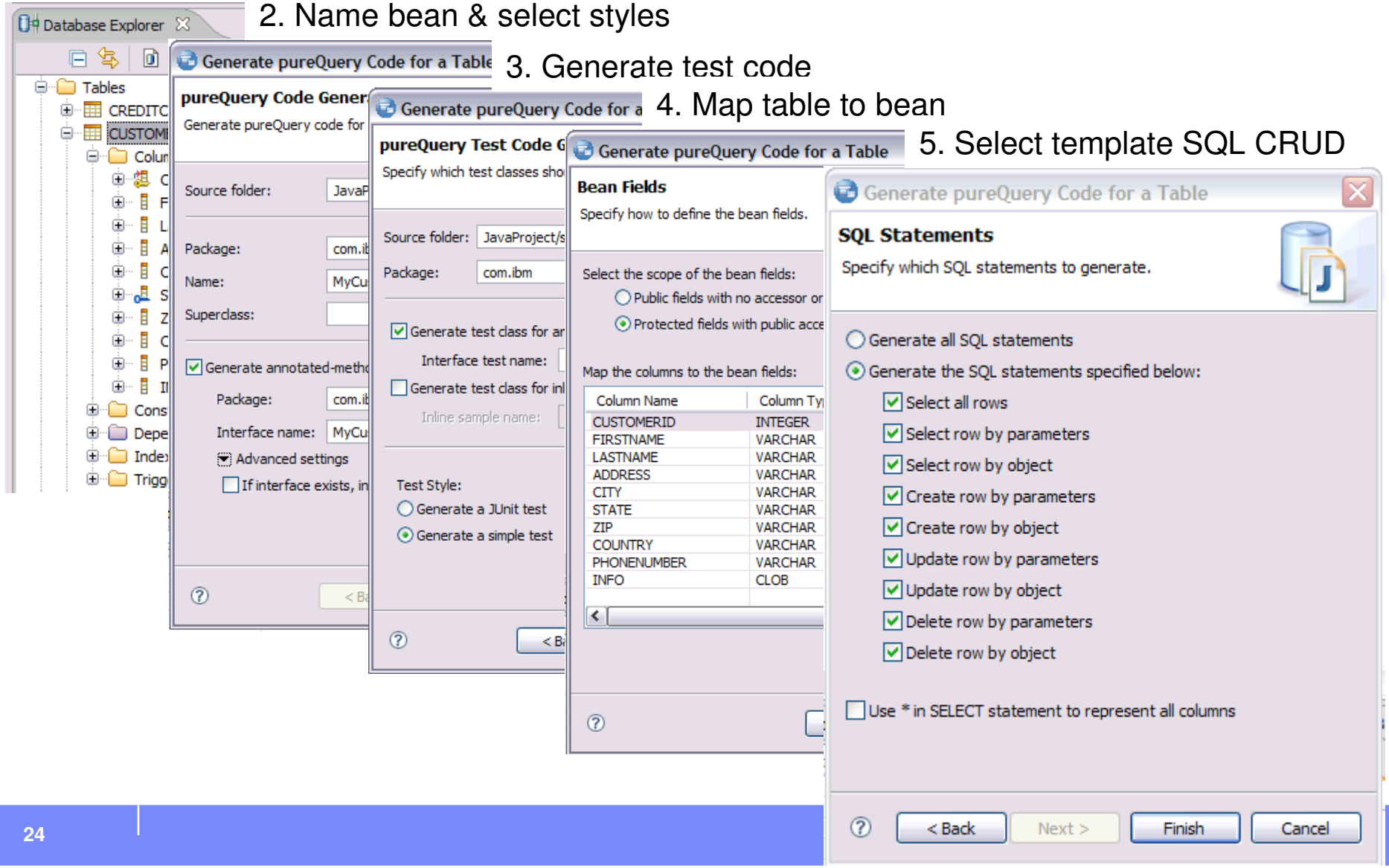
Map the columns to the bean fields:

Column Name	Column Type	Field Name	Field Type
CUSTOMERID	INTEGER	customerId	int
FIRSTNAME	VARCHAR	firstName	String
LASTNAME	VARCHAR	lastName	String
ADDRESS	VARCHAR	address	String
CITY	VARCHAR	city	String
STATE	VARCHAR	state	String
ZIP	VARCHAR	zip	String
COUNTRY	VARCHAR	country	String
PHONENUMBER	VARCHAR	phoneNumber	String
INFO	CLOB	information	String

< Back | Next > | Finish | Cancel

Java Data Access in 5 simple steps

1. Select tabl
2. Name bean & select styles
3. Generate test code
4. Map table to bean
5. Select template SQL CRUD



More Visibility and Control of Application SQL

- **Capture SQL**
- **Share, review, and optimize SQL**
- **Revise and validate equivalency**
- **Bind for static execution or run dynamically**
- **Restrict SQL to eliminate SQL injection**

ITPRO FIT FOR BUSINESS

IT PRO has been watching and charting the progress of what is one of the biggest and most high profile web security threats of this year - the SQL injection.
By Asavin Wattanajantra, 4 Aug 2008 at 11:55

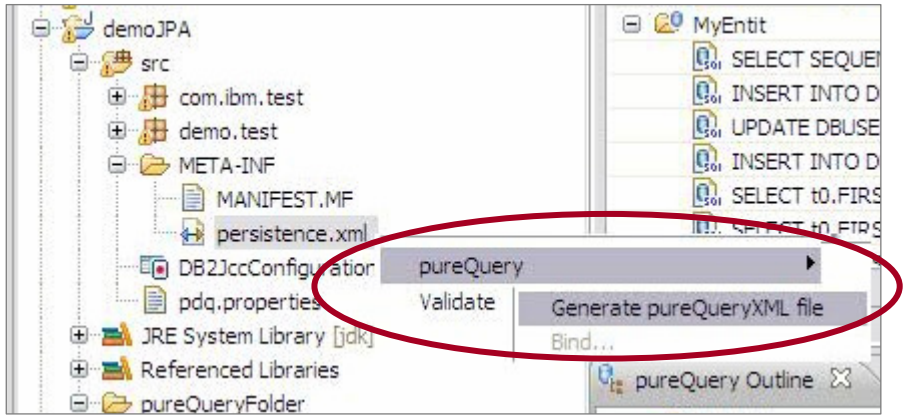
Capture → Review → Optimize → Revise → Restrict

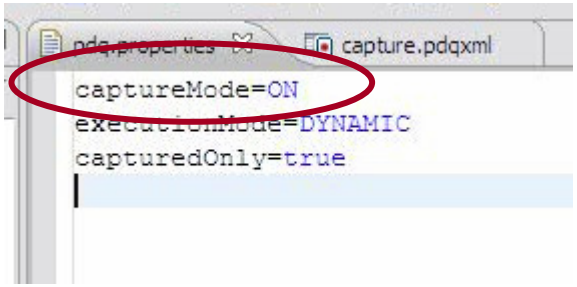
Capture Application SQL: At Development or Later

Capture → Review → Optimize → Revise → Restrict

Three methods

1. Use pureQuery API
2. Use JPA and generate the pureQuery file
3. Set captureMode=ON and execute the program

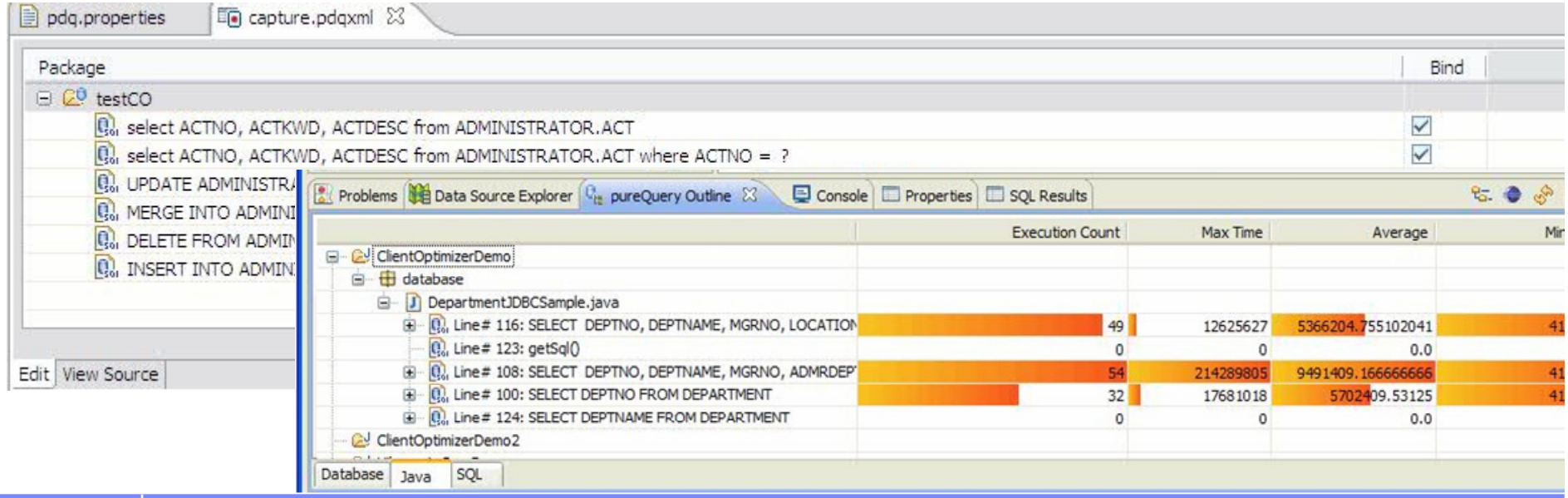




Visualize Application and SQL Metadata

Capture → **Review** → Optimize → Revise → Restrict

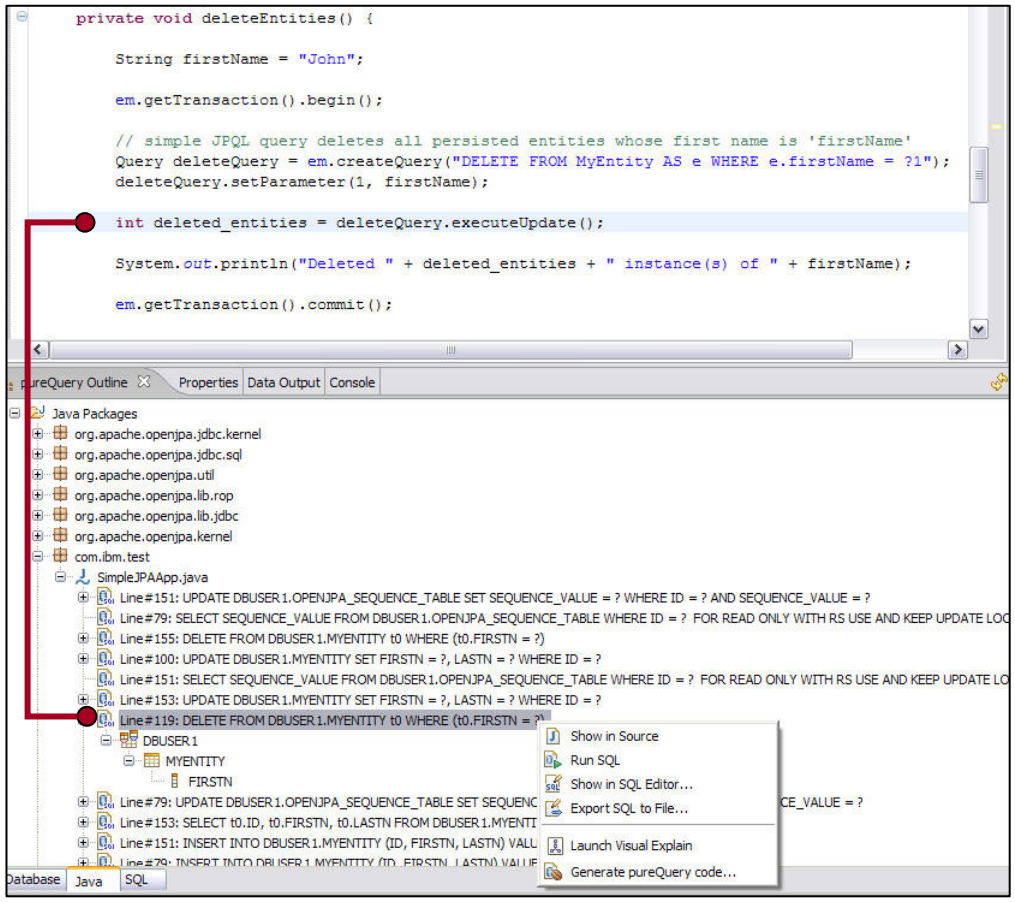
- Review the captured SQL
- View metrics about execution frequency and duration
- Share captured SQL with DBA



pureQuery Outline

Speed up problem isolation for developers – even when using frameworks

- Capture application-SQL-data object correlation (with or without the source code)
- Trace SQL statements to using code for faster problem isolation
- Enhance impact analysis identifying application code impacted due to database changes
- Answer “Where used” questions like “Where is this table used within the application?”
- Use with modern Java frameworks e.g. Hibernate, Spring, iBatis, OpenJPA



Problem Determination

Correlate Package and SQL With Captured Metadata

Java - IBM Data Studio Developer

File Edit Navigate Search Project Data Run Window Help

Package Explorer | Hierarchy

- CLientOptimization
  - src
  - JRE System Library [jdk]
  - db2jcc_license_cisuz.jar - C:\Progr
  - db2jcc.jar - C:\Program Files\IBM\S
  - pdq.jar
  - pdqmgmt.jar
  - pureQueryFolder
    - capture.pdqxml
    - capture.pdqxml.or
    - Default.bindProps
    - Default.genProps

Session A - [24 x 80]

File Edit View Communication Actions Window Help

```
 _                ZSQL     VTM     02       V410./C DB1S 08/11/08 15:48:30   5
> A-THREAD DETAIL B-LOCK COUNTS C-LOCK WAITS     D-LOCKS OWNED  E-GLOBAL LOCKS
> *-CURRENT SQL   G-SQL COUNTS  H-DISTRIBUTED    I-BUFFER POOL  J-GROUP BP
> K-PACKAGES      L-RES LIMIT   M-PARALLEL TASKS N-UTILITY      O-OBJECTS
> P-CANCEL THREAD Q-DB2 CONSOLE R-DSN ACTIVITY   S-APPL TRACE   T-ENCLAVE
> U-LONG NAMES    V-SQL PA
================================================================================
>                          SQL CALL BEING EXECUTED
 PLAN
+ Thread:  Plan=DISTSERV  Connid=SERVER   Corrid=db2jcc_appli Authid=DBA031
+ Dist  :  Type=DATABASE ACCESS, Luwid=G9274097.H546.C2D458DA7274=15806
+      SQL call is active, call information is as follows :
+
+      Thread Status   = IN-DB2          SQL Request Type     = STATIC
+      Total SQL Reqs =        93        SQL Call Type        = FETCH
+      SQL DBRM Name   = Order_de        SQL Statement Number = 00001
+      Collection ID   = NULLID
+
+      DECLARE DB2JCCCURSOR1 CURSOR FOR SELECT ORDER_DETAIL_CODE, ORDER_NUMBER,
+       SHIP_DATE, PRODUCT_NUMBER,  PROMOTION_CODE, QUANTITY, UNIT_COST, UNIT_P
+      RICE, UNIT_SALE_PRICE  FROM GOSALES.ORDER_DETAILS FOR READ ONLY
```

Database Explorer

- Connections

Console | Properties | pureQuery Outline

- DB2 Packages
  - Order_de
    - SELECT ORDER_DETAIL_CODE, ORDER_NUMBER, SHIP_DATE, PRODUCT_NUMBER, PROMOTION_CODE, QUANTITY, UNIT_COST, L

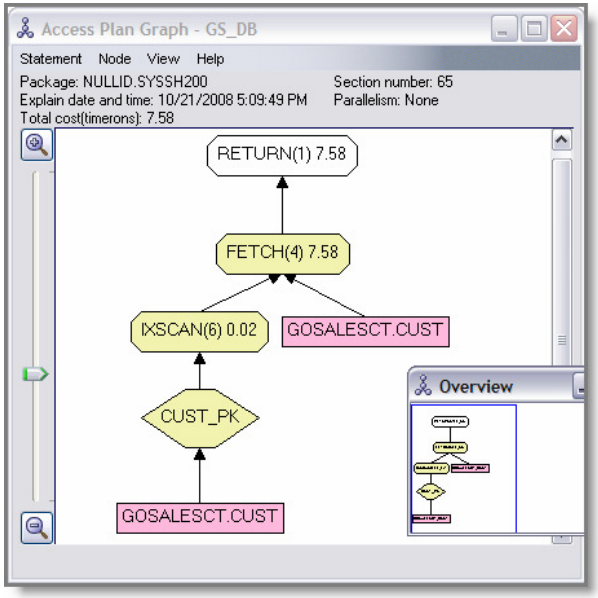
Database | Java | SQL

0 items selected

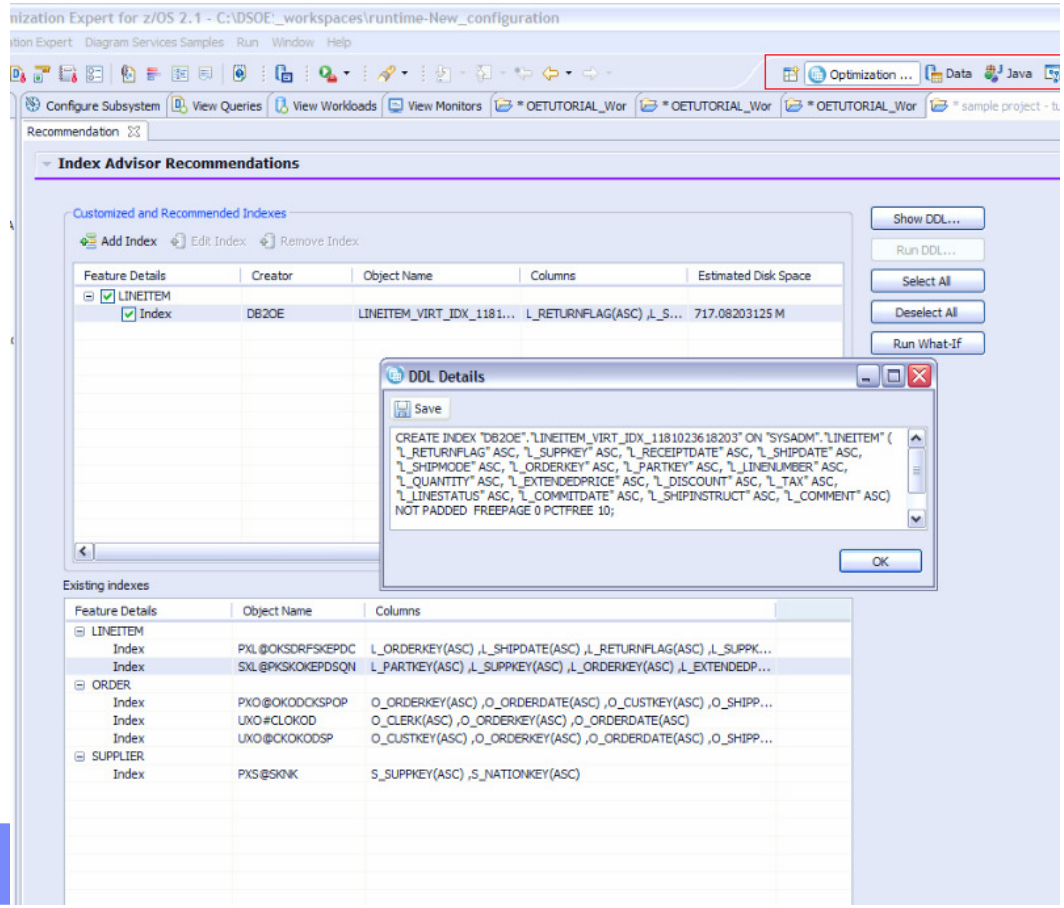
Optimize SQL

Capture → Review → **Optimize** → Revise → Restrict

- Launch Visual Explain



- Copy SQL to Optimization Expert



Revise SQL Without Modifying the Application

Capture → Review → Optimize → Revise → Restrict

capture.pdqxml

Package

testCO

select ACTNO, ACTKWD, ACTDESC from ADMINISTRATOR.ACT

select ACTNO, ACTKWD, ACTDESC from ADMINISTRATOR.ACT where ACTNO = ?

UPDATE ADMINISTRATOR.ACT SET ACTNO = ? , ACTKWD = ? , ACTDESC = ? WHERE ACTN

MERGE INTO ADMINISTRATOR.ACT AS t USING table(VALUES(CAST(? AS SMALLINT), CAST(... AR(20)))) AS s(ACTNO, AC

DELETE FROM ADMINISTRATOR.ACT WHERE ACTNO = ?

INSERT INTO ADMINISTRATOR.ACT (ACTNO, ACTKWD, ACTDESC) VALUES (? , ? , ?)

Copy

Edit Statement

Delete Statement

Open Default.bindProps

Edit or delete SQL

capture.pdqxml

Package

testCO

select ACTNO, ACTKWD, ACTDESC from ADMINISTRATOR.ACT

select ACTNO, ACTKWD, ACTDESC from ADMINISTRATOR.ACT where ACTNO = ?

select ACTNO, ACTKWD, ACTDESC from ADMINISTRATOR.ACT where ACTNO = ?

UPDATE ADMINISTRATOR.ACT SET ACTNO = ? , ACTKWD = ? , ACTDESC = ? WHERE ACTNO = ?

MERGE INTO ADMINISTRATOR.ACT AS t USING table(VALUES(CAST(? AS SMALLINT), CAST(? AS CHAR(6)), CAST(? AS VARCHAR(20)))) AS s(ACTNO, AC

DELETE FROM ADMINISTRATOR.ACT WHERE ACTNO = ?

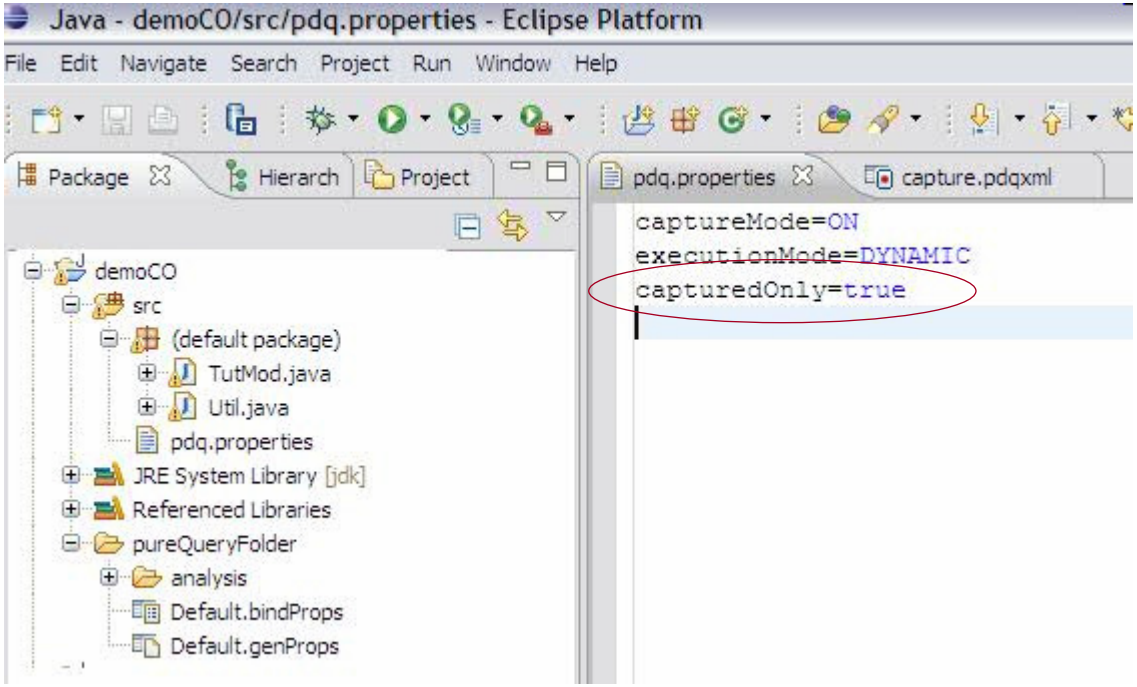
INSERT INTO ADMINISTRATOR.ACT (ACTNO, ACTKWD, ACTDESC) VALUES (? , ? , ?)

Tooling validates equivalent SQL
Shows replaced

Eliminate SQL Injection

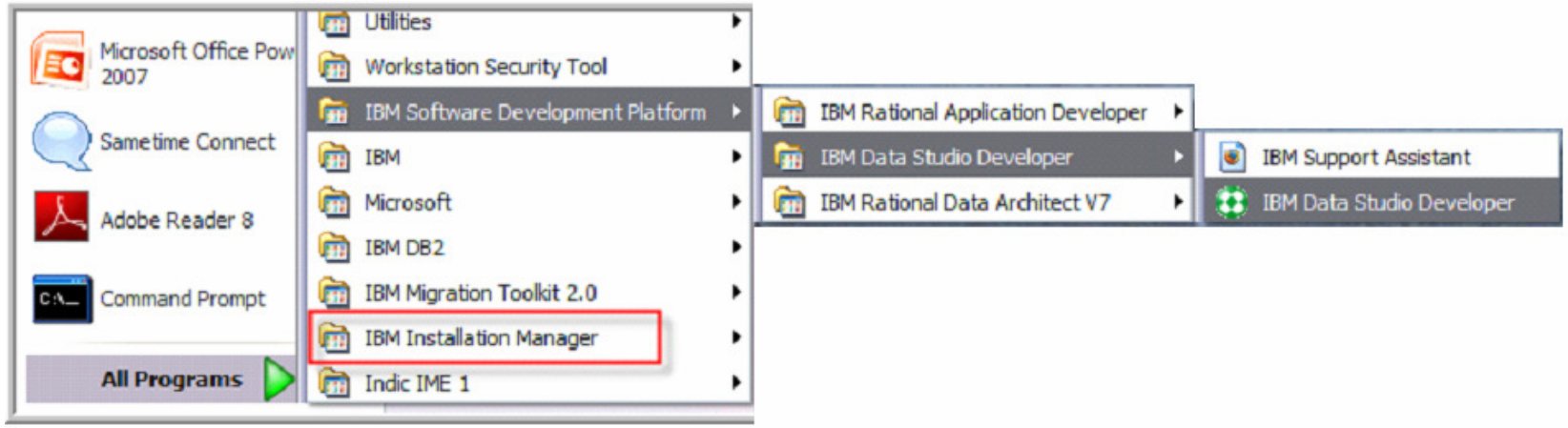
Capture → Review → Optimize → Revise → Restrict

- Restrict SQL execution to only those statements captured
- Set capturedOnly=true in pdq.properties
- pureQuery Runtime looks for it it in the classpath



Extend Rational Application Developer Capabilities

- **Integrate pureQuery development into existing Eclipse tools environment**
  - Contextual SQL assistance
  - Data access layer
  - Unit test generation
  - Impact analysis and SQL traceability
  - Extended database object support
- **Common workspace allows you to be more productive**
- **Shell share with Rational Software Delivery Platform v7.5**

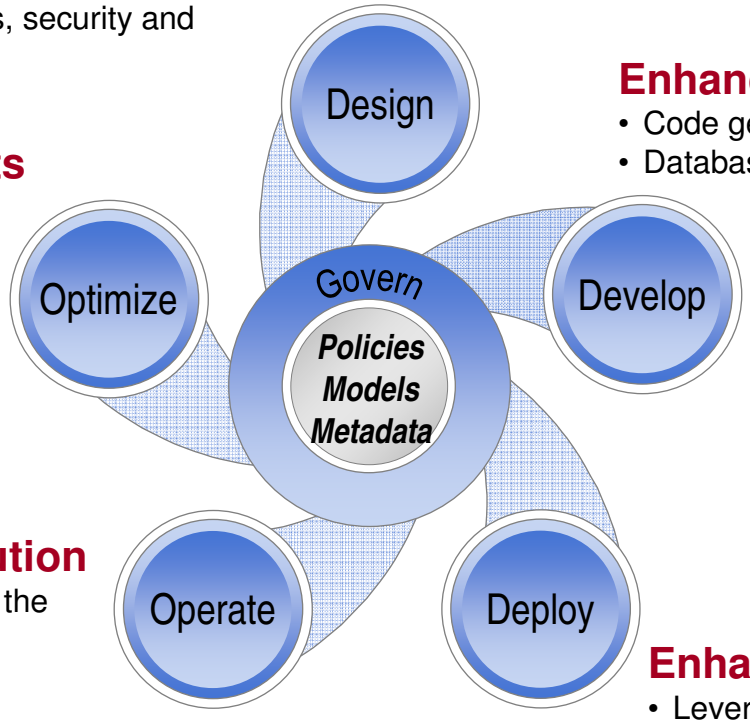


Optimize for WebSphere and DB2 with pureQuery

Capture metadata from existing applications
- Capture from JPA without executing
- Derive performance, costs, security and manageability value

Jump start application design
- Generate SQL and Code from Database Objects
- Setup basic DAO Pattern

Reduce HW and SW costs
- Up to 42% lower CPU/Trans
- Move workload to zIIP and zAAP



Enhance development productivity
- Code generation, content assist
- Database aware, Java SQL Editor

Simplify impact analysis
- Categorize by Java, SQL, Database, Packages, track back to line of code

Replace SQL without changing the source
- Editor validates equivalency

Focus tuning efforts
- Find and sort by query elapsed time from Java

Speed up problem resolution
- Trace SQL back to line of code in the application

Enhance performance
- Leverage best practices, automatically for JPA
- Use static execution, automatically for JPA
- Lock in access plans for consistent performance

Prevent SQL Injection
- Lock down SQL for dynamic or static execution

Reduce security exposure
- Grant access to queries, not tables

Thank You for Joining Us today!

- **Go to www.ibm.com/software/systemz to:**
  - Replay this teleconference
  - Replay previously broadcast teleconferences
  - Register for upcoming events
- **For more on pureQuery**
  - On the Web: http://ibm.com/software/data/studio
  - IBM Data Studio: The Big Picture
    - http://www.ibm.com/developerworks/db2/library/techarticle/dm-0807hayes/
  - pureQuery overview: The Easy Way to Quick Data Access
    - http://db2mag.com/story/showArticle.jhtml?articleID=202400140
  - IBM Data Studio pureQuery Runtime for z/OS Performance
    - http://www.ibmdatabasemag.com/story/showArticle.jhtml?articleID=208802229
  - Webcasts and more in the Data Studio Community
    - http://www.ibm.com/developerworks/spaces/datastudio
  - Complementary Proof of Technology
    - Talk to your IBM Sales Rep to register for a 1-day proof of technology with introduction and hands-on labs