The International Journal of Digital Accounting Research
Vol. 12, 2012, pp. 121 – 144
ISSN: 1577-8517

Internal Control Computerization for Derivatives

Seokjoo Andrew Chang. State University of New York at Albany**.** USA**.** schang@albany.edu

Guy Fernando. State University of New York at Albany**.** USA**.** gfernando@albany.edu

Mohamed E. Hussein. University of Connecticut. USA. mohamed.Hussein@business.uconn.edu

Kinsun Tam. State University of New York at Albany. USA. tam@albany.edu

Abstract. Derivatives were at the center stage of many financial scandals, incidents of massive trading losses, as well as the subprime mortgage crisis of 2008. Monitoring of derivatives transactions is desperately needed. As derivatives trading systems are computerized, management and accountants need to acquire IT knowledge and skills necessary to evaluate computerized internal controls on derivatives. To prepare management and accountants for derivatives control monitoring, this paper discusses implementation of computerized internal controls in relational database management systems over the life cycle of derivatives transactions.

Keywords: derivatives; computerized internal control; database constraints; relational database management system; structured query language

1. INTRODUCTION

Computerizing internal control for derivatives transactions is an important topic in the current business environment. Derivatives transactions are different from other business areas because of the sheer size, high risk, and electronic trading of derivatives. Derivatives are economically important. According to the International Swaps and Derivatives Association and the World Federation of Exchanges (2012), the total notional amount outstanding of interest rate, credit, and equity derivatives at June 30, 2010 was $467 trillion, which was more than 8 times the $55 trillion of stocks outstanding at the end of 2010. Traded with high

DOI: 10.4192/1577-8517-v12_5

Submitted June 2012
Accepted August 2012

leverage, derivatives may expose trading institutions and even the economy to huge risks. Unlike in other business areas, a single employee mishandling derivatives can cause multibillion dollar losses, sometimes even toppling the employer firms (e.g. bankruptcy of the Barings Bank). Risks in derivatives are exacerbated by the fact that derivatives are traded electronically which requires pervasive and real-time internal controls. Hence, management and accountants need to have a good understanding of computerized internal controls to monitor derivatives activities so as to take the necessary action when violations are detected.

Computerizing internal controls is an interdisciplinary topic. This paper combines techniques from both accounting and computer science to help management and accountants develop capabilities in using computerized internal controls. Section 2 discusses the role of internal control and computerized internal control. Section 3 reviews the current literature. Section 4 discusses the risks of derivatives. Section 5 introduces relational databases as a platform where computerized internal controls are implemented (Chen, 1976). Section 6 discusses the implementation of computerized internal controls for derivatives. Concluding observations, limitations, and future research opportunities are presented in Section 7.

2. INTERNAL CONTROL AND COMPUTERIZED INTERNAL CONTROL FOR DERIVATIVES

Internal controls are safeguards against material risks of errors and fraud. Failures in internal controls may yield disastrous results. Responding to problems and risks surrounding the use of derivatives (e.g. massive losses by Procter & Gamble in 1994, Orange County in 1994, Barings Bank in 1995, etc.), the Committee of Sponsoring Organizations of the Treadway Commission (COSO, 1996) published "Internal Control Issues in Derivatives Usage" in 1996. In this document, COSO encouraged management to strengthen internal control systems for derivatives activities, and helped firms apply the time-proven "Internal Control - Integrated Framework" to derivatives usage. Unfortunately, colossal internal control failures continued to plague prominent institutions. Derivatives were responsible for twelve more multibillion dollar trading losses between 1996 and 2012. In particular, French bank Societe Generale's arbitrage trader, Jerome Kerviel, traded

European stock futures beyond authorized limits between 2006 and 2008, causing a staggering $7.2 billion loss. On Sep 16, 2011, London Police charged Kweku Adoboli, a director of UBS' equities trading team, with fraud and false accounting. Adoboli's unauthorized trades of equity derivatives cost UBS $2 billion in losses.

In both Societe Generale and UBS, internal control systems were in place before scandals occurred. However, Kerviel breached multiple layers of controls, and carefully closed his unauthorized trades just two or three days before the trades' timed controls would trigger notice from the bank's internal control system (New York Times Jan 25 2008; Wikipedia Jan 2, 2012). Societe Generale acknowledged having failed to act on at least 74 internal trading alerts dating from mid-2006 (New York Times Oct 5, 2010). Likewise, Adoboli allegedly overrode UBS's internal controls (New York Times Sep 16, 2011). UBS's computer system detected Adoboli's unauthorized trading activities and issued a warning, which however was not sufficiently investigated or appropriately acted upon (Computerworld Oct 6, 2011). These two cases suggest that having skilled systems professionals to implement computerized internal control systems is just one part of the equation. Computerized systems must be constantly monitored. Internal controls must be reviewed for loopholes, and warnings from internal control systems must be handled seriously. Managers monitoring derivatives exposures must have good understanding of operational and technical aspects of computerized internal controls to know when and how to react to warning signals from computer systems.

Not all managers and accountants are prepared to review, monitor, and respond to computerized internal controls on derivatives. The complexity of derivatives instruments is the first challenge. For instance, Procter & Gamble committed to interest rate swaps in 1994 without fully understanding the nature of risk exposures. Not prepared to monitor and manage risks of exotic derivatives, Procter & Gamble's management could not but allow losses to quickly snowball. In a well noted Congressional testimony in 1994, George Soros conceded that for derivatives, "the risks involved are not always fully understood even by sophisticated investors, and I am one of them." Second, since derivatives are electronically traded, much of the corresponding internal controls involve high level information technologies and computerization. By discussing computerized

internal controls over the life cycle of derivatives transactions, this paper is of value to management and accountants (including external and internal auditors), and uniquely contributes to the literature on derivatives and internal control.

Conventional wisdom suggests human beings are inclined to stay away from unfamiliar situations. Adopting the traditional black-box view of the computer, many managers and accountants are not prepared to actively engage computerized internal control systems. As they turn away from such systems, as in the cases of Societe Generale and UBS, the effectiveness of these systems is drastically undermined. For computerized internal control systems to accomplish their intended mission, the people monitoring these systems and watching out for warning signals must abandon the traditional black-box view of the computer.

The Societe Generale and UBS scandals are sending a new warning to management and accountants. As a far cry from management's and accountants' computer aversion, derivatives-fraud perpetrators are becoming decidedly computer savvy. Colleagues described Kerviel as a computer genius who purportedly burrowed into Societe Generale computer and spent many hours to evade controls (The Wall Street Journal Jan 25, 2008; The Telegraph Jan 25, 2008). Adoboli studied computer science at the University of Nottingham, and subsequently gained knowledge of UBS's computer systems and protocols from back office work (New York Times Sep 15, 2011 and Sep 16, 2011). To catch up with the nemesis, management and accountants must update their training on computer systems and computerized internal controls.

Management and accountants are responsible for evaluating internal controls, including computerized internal controls. The Sarbanes-Oxley Act of 2002 requires management of public companies to assess the effectiveness of internal controls, and accountants to attest to the management's assessment. For businesses using computerized accounting information systems, management and accountants need an appropriate understanding of the underlying technologies to evaluate computerized internal controls to comply with SEC Rule 33-8238. Likewise, professional guidelines in accounting highlight the role of information technology (IT) in internal controls, and recommend strengthening accountants' understanding of IT controls. SAS No. 94 acknowledges that a significant amount of information supporting financial statement assertions is electronically initiated,

recorded, processed, and reported by IT systems with automated procedures, and requires the auditor to obtain sufficient knowledge of information systems to understand such procedures. The Canadian Institute of Chartered Accountants (CICA 2007), the AICPA (2009), the Institute of Internal Auditors (IIA 2009), the International Federation of Accountants (IFAC, 2010), and the Information Systems Audit and Control Association (ISACA, 2012) all call on accountants and auditors to be proficient with computerized controls and/or the technologies underlying internal controls in computer based business systems.

In response to the above calls and to contribute to the literature on control systems for derivatives, this paper discusses implementation of computerized internal controls in relational database management systems (RDBMS) over the life cycle of derivatives transactions. These controls include preventive controls and detective controls, as well as controls for profit-oriented traders and hedge-oriented end users. Internal controls, no matter how well designed, are understood to have inherent limitations and may become inadequate because of changes in conditions. COSO (1996) addresses these shortcomings by suggesting internal control monitoring to help ensure "that internal control continues to operate effectively." Continual monitoring of computerized internal controls over derivatives transactions is a human issue. Reviewing internal control systems for loopholes and acting on signals from these systems are the responsibilities of management and accountants, who need appropriate technical training. To target these audience groups, the discussions in this paper are intentionally simple, and use of technical jargon is minimized. By helping management and accountants strengthen internal controls over derivatives transactions, this paper hopes to contribute to the defense against future derivatives scandals.

3. LITERATURE REVIEW

Prior research on computerized internal controls has been sporadic. Krishnan *et al.* (2005) represent internal controls in an extended BPMN (business process modeling notation), which potentially supports automatic generation of executable code for accounting systems and internal controls. Since Krishnan *et al.* (2005) leave the work on code generation to future research, internal controls expressed in extended BPMN remain non-executable. Databases offer a more mature platform than BPMN for implementing computerized internal controls. Agrawal

et al. (2006) advocate using database technologies to automate internal control processes, but offer no SQL coding to illustrate such uses. This paper extends Agrawal *et al.* (2006) by implementing computerized internal controls throughout the life cycle of derivatives transactions in SQL. Some of these controls require management to specify thresholds to filter out anomalies for further investigation. When management cannot determine appropriate thresholds, one solution is to call upon the unsupervised learning algorithm of cluster analysis to identify abnormal transactions. Thiprungsri and Vasarhelyi (2011) propose using cluster analysis to help auditors detect fraud, and report on applying cluster analysis to pinpoint anomalies in group life insurance claims. Therefore, the computerization of internal controls illustrated in this paper can be integrated with cluster analysis to supplement management's judgment.

As a topic straddling the disciplines of accounting and computer science, computerized internal control tends to be overlooked by both sides. Accountants are trained to be professionals in business processes and internal controls, and in recording, classifying, and summarizing transactions but not computer programming. Understandably, the accounting literature stays away from technical details, and omits computerized internal controls in RDBMS. On the other hand, the computer science discipline focuses on programming and software engineering, but not necessarily in the business and accounting context. Computer scientists traditionally receive no professional training in internal control, not to mention derivatives. The computer science literature, with very few exceptions such as Agrawal *et al.* (2006), almost never refers to "internal control". The database literature typically discusses "constraints" (defined in Section 5) in mathematical symbols and programming notations, but rarely in an accounting context. Ullman and Widom (2007), for instance, define database constraints as "Boolean value functions whose value is required to be true", and discuss constraints for a database about movies and movie stars, which is devoid of practical accounting context. Computerized internal controls, as a research topic, benefit from advances in accounting and computer science. The internal control roles of database systems deserve more attention from computer science and accounting researchers. This paper augments both the accounting and the computer science literature by exposing an overlooked research topic intersecting the two disciplines.

4. RISKS OF DERIVATIVES

Derivatives are risky because of their high leverage. Many derivatives require little to no initial cost, allowing market participants to commit to large contracts (i.e. billions of dollars in notional principal amounts) with massive gain or loss exposures. Derivatives market participants include both profit-oriented traders and hedge-oriented end users. Profit-oriented traders are sophisticated market participants (e.g. arbitragers, brokers/dealers, and speculators) who trade derivatives to make a profit. Arbitragers balance their derivatives positions with opposite positions on the underlying assets, and seek to profit from any price differences. Brokers and dealers do not maintain positions on derivatives but mediate positions for customers to earn commissions or spreads. Arbitragers and brokers/dealers, who are often units within investment banks, have minimal net exposures unless they deviate from their expected roles. If arbitragers do not balance their positions or if brokers and dealers take positions on derivatives, they essentially become speculators who are exposed to risk. Arbitrager and brokers/dealers not intending to engage in speculation can implement internal controls to limit speculative trading. Speculators maintain derivatives positions to bet on and hopefully profit from a given direction of market movement. Speculation is a regular business of investment banks and hedge funds which nonetheless need internal controls to contain speculation-related risk exposures to a manageable level.

End users include industrial firms, retirement funds, insurance companies, etc. who use derivatives to hedge but are not necessarily sophisticated market participants. End users using derivatives to fully hedge other assets and liabilities have little net exposure because the gains and losses from derivatives induced by market movements should offset the losses and gains from these assets and liabilities. However, if end users acquire a naked position on derivatives, they become speculators and are exposed to risk. Internal controls help end users guard against speculative trading.

Derivatives scandals in the last 20 years have crippled many end users, hedge funds, and investment banks and their arbitrage and broker/dealer units. Internal controls are needed to assure that derivatives trading is conducted within an acceptable level of risk exposure. A purely manual internal control system is

unlikely to be effective because derivatives transactions with huge gain/loss implications can happen electronically in milliseconds, which is much quicker than the human being's response time. Computerized internal controls electronically prevent and detect violating transactions, and are crucial to risk management for all derivatives market participants.

5. RELATIONAL DATABASE AND INTERNAL CONTROL

Databases support numerous business applications and are widely used commercially. Many managers and accountants interact with database systems in the work environment as users or handlers of data. Today's databases are almost always based on the relational model (Ullman and Widom, 2007). The model is widely accepted because its basic modeling concept -- the relation -- is simple to understand and use. The relational model represents data in two-dimensional tables (a.k.a. relations). A row of data is known as a tuple. A table column is called an attribute. Database systems based on the relational model are known as RDBMSs (relational database management systems). In RDBMS, the computer language for defining database tables and manipulating data is called SQL (structured query language). SQL supports constraints, assertions, and triggers (collectively known as "active elements") which are used to enforce business rules (i.e. internal controls) and guard against violating transactions.

A constraint defines the conditions under which data is valid. RDBMS calls upon the constraint whenever the elements being constrained are changed. Modifications (including insertions, deletions, or updates) to the database that violate the constraint will be rejected. A referential integrity constraint (a.k.a. foreign key constraint) requires that a value referred to by some object actually exist in the database (e.g. the counterparty named in the transaction must appear in the table of approved counterparties). An attribute-based constraint expresses the requirement that the value of an attribute must be drawn from a specific set or range of values (e.g. net assets of a counterparty must be above $500 million). A tuple-based constraint specifies a requirement over multiple attributes of a tuple (e.g. the loss attribute of the Trader table must not exceed the loss limit attribute).

Some business rules are too general for constraints. For instance, the business rule that "a trader's rank determines her loss limit such that two traders at the same rank should have the same loss limit" is an example of functional dependency,

which is implemented as an assertion. This assertion then enforces the rule that "rank functionally determines loss limit". Similarly, business rules restricting multiple tables can be implemented as assertions (or alternatively as tuple-based constraints). As a more general version of constraint, an assertion can replace any constraint discussed in this paper. An assertion is evaluated whenever any table named in the assertion is changed. The conditions specified in an assertion must always be true. Any modification whatsoever that violates the assertion will be rejected.

While constraints and assertions prohibit violating transactions, triggers simply perform follow-up actions. Triggers are activated when certain pre-specified database events (i.e. insert, delete, or update) occur. An activated trigger first tests a condition. If the condition is satisfied, the action specified in the trigger will be performed by the RDBMS. A trigger is a convenient mechanism to implement a certain class of internal controls (e.g. generating an exception report when total trading losses exceed a predetermined threshold). To help management and accountants develop proficiency in computerized internal controls, the next section will discuss internal controls for derivatives implemented in SQL as constraints, assertions, or triggers.

6. COMPUTERIZED INTERNAL CONTROLS FOR DERIVATIVES

Internal controls are needed throughout the life cycle of a derivatives transaction. This section is devoted to implementing computerized internal control at various stages of this life cycle including (1) before trading, (2) at the point of trading, and (3) subsequent to trading. No matter how well-conceived, internal controls are subject to inherent limitations and cannot provide absolute assurance regarding the achievement of objectives. The reader is reminded that the controls discussed below are intentionally simplified for non-technical managers and accountants, and are not meant to be comprehensive. These controls and their associated life cycle stages are summarized as follows:

- Internal Controls before Trading
    1. Screening counterparties of derivatives transactions.
    2. Setting loss limits on each trader.
    3. Defining authorized derivatives products.

- Internal Controls at the Point of Trading
    1. Trading only with approved counterparties.
    2. Suspending any trader who has exceeded the loss limit.
    3. Checking trader authorization on product type.
    4. Limiting the exposure to each category of product risk.
    5. Limiting the exposure to each category of product risk (for derivatives end users).
    6. Monitoring the daily frequency of trades (for derivatives end users).
- Internal Controls Subsequent to Trading
    1. Monitoring warning signals on total loss.
    2. Monitoring the gain of each trader.

A relational database with five tables will illustrate the computerization of the above internal controls over derivatives trading. To conserve space and enhance clarity, the tables are intentionally simplistic. All monetary values in these tables are in millions. The `Counterparty` table keeps trading counterparty attributes such as firm, address, id numbers, and net assets (Table 1). The `AuthorizedProduct` table records the trader's employee ID and the products that the trader is allowed to trade (Table 2). The `Trader` table records the trader's employee ID, year-to-date gains earned or losses sustained (including both realized and unrealized components), rank, and the loss limit beyond which the trader must suspend trading (Table 3). The `Trade` table keeps track of trade attributes including trade ID, counterparty id, trade date, notional principal amount of the trade, traded instrument, trader's employee ID, and direction of the position (Table 4). Finally, the `ExceptionReport` table has only the `message` attribute (Table 5).

firm	address	id	netAsset
Morgan Equities	5 Main St	c30011	830
Goldman Investments	10 West St	c30022	11300

Table 1. Counterparty Table

employId	productId
e633	euroFutures
e109	euroFutures
e089	tBondFutures
e089	euroFutures

Table 2. AuthorizedProduct Table

employeeId	gainLoss	Rank	lossLimit
e633	-12	Senior	10
e109	0	Chief	50
e089	62	Chief	50

Table 3. Trader Table

tradeId	cpId	tradeDate	notional Principal	Instrument	eId	position
t3988	c30011	nov26	880	tBondFutures	e089	short
t3989	c30022	nov29	3000	sp500Futures	e633	short
t3990	c10022	nov29	1500	tBondFutures	e089	long

Table 4. Trade Table

Message
Trader e089 gained 62 million from derivatives trading.

Table 5. ExceptionReport Table

The SQL statements to define and create these tables are given below (Table 6). Although internal controls can be specified in SQL statements when creating tables, this paper will separate table creation from internal control implementation. This way, subsequent discussion of each internal control will entail only those SQL codes necessary for implementing the internal control in focus. This approach helps avoid unnecessary repetition of table-definition statements when each table may be associated with multiple internal controls.

```
CREATE TABLE Counterparty
(
   firm CHAR(64),
   address CHAR(128),
   id CHAR(6),
   netAsset REAL
);

CREATE TABLE AuthorizedProduct
(
   employId CHAR(4),
   productId CHAR(16)
);

CREATE TABLE Trader
(
   employeeId CHAR(4),
   gainLoss REAL,
   rank CHAR(32),
   lossLimit REAL
);
```

Table 6. Defining and creating tables

```
CREATE TABLE Trade
(
   tradeId CHAR(5),
   cpId CHAR(6),
   tradeDate CHAR(5),
   notionalPrincipal REAL,
   instrument CHAR(16),
   eId CHAR(4),
   position CHAR(8)
);

CREATE TABLE ExceptionReport
(
   message CHAR(512)
);
```

Table 6. Defining and creating tables (continuation)

Each `CREATE TABLE` statement specifies the name of the table to be created and a list of attributes each followed by the data type. For instance, the first statement defines the `Counterparty` table as having `firm`, `address`, `id`, and `netAsset` as attributes. `CHAR(64)` denotes character data type allowing up to 64 spaces. The `REAL` data type is for decimal values. SQL is case insensitive. The use of upper case for SQL keywords and lower case for variable names is an optional convention simply to improve clarity of presentation. Similarly, use of extra spaces and line breaks is optional.

Internal Controls before Trading

1. Screening derivatives transaction counterparties. Derivatives expose transacting firm to immense counterparty risk (e.g. default risk). One way to mitigate this risk is to trade only with counterparties of solid financial standing. Counterparty firms with weak financial standing can be filtered out with proper internal controls over financial credentials. For instance, the internal control that each counterparty firm's net assets must be at least 500 million dollars is implemented as an attribute-based constraint, named `screenCounterparty`, in SQL (Table 7).

```
ALTER TABLE Counterparty ADD CONSTRAINT screenCounterparty
   CHECK (netAsset >= 500);
```

Table 7. Attribute-based constraint to screen counterparties

The above SQL statement instructs the RDBMS to alter the Counterparty table by adding a constraint to check that every record in the table must have at least $500 million of net assets. This attribute-based constraint is checked whenever the `netAsset` attribute is modified.

Moreover, some derivatives scandals in the past involved rogue traders hiding unauthorized transactions (e.g. those involving unauthorized products or beyond authorized limits) by trading with many counterparties. Maintaining a list of approved counterparties and forcing the trader to trade only with counterparties on this list will mitigate the risk of such scandals.

2. Setting loss limits on each trader. Derivatives scandals could result in staggering losses when traders seek to recoup earlier losses by doubling subsequent bets (e.g. Olympus Corporation's Mori and Yamada in 2011, UBS's Adoboli in 2011, Allied Irish Bank's John Rusnak in 2002, and Barings Bank's Nick Leeson in 1995). Hence, firms trading derivatives must carefully determine the appropriate loss limit beyond which the trader is to be suspended from trading. Management may decide to set loss limits based on the rank of the trader. For instance, junior traders are given a loss limit of $1 million, senior traders $10 million, chief traders $50 million, etc. Under this business rule, the rank attribute is said to functionally determine the loss limit attribute because whenever two rows agree on rank, they must also agree on loss limit. This functional dependency is too general to be implemented as a constraint, but can be implemented as an assertion, named `rankDetermineLimit`, as shown below (Table 8).

```
CREATE ASSERTION rankDetermineLimit CHECK
(NOT EXISTS
   (SELECT * FROM Trader AS T1 CROSS JOIN Trader AS T2
     WHERE T1.rank = T2.rank and
     T1.lossLimit <> T2.lossLimit
   )
);
```

Table 8. Assertion to determine loss limit from rank

This assertion is evaluated whenever the Trader table is modified, and rejects any modification that causes two rows with the same value on the rank attribute to disagree on the `lossLimit` attribute. The two `AS` operators make two temporary

copies T1 and T2 of the Trader table. CROSS JOIN creates new rows by concatenating each row from T1 with each row from T2. The SELECT operator looks for new rows where T1's rank attribute agrees with T2's rank attribute, but T1's lossLimit attribute disagrees with T2's lossLimit attribute. The NOT EXISTS operator requires that the SELECT operator find no satisfying row.

3. Defining authorized derivatives products. Some derivatives scandals involved trading unauthorized derivatives (e.g. Citic in 2008, Barings Bank 1995, and Proctor and Gamble in 1994). Control over allowable derivatives types needs to be enforced especially for end users who, unlike investment banks, arbitragers, derivatives brokers/dealers, and speculators, may not have detailed knowledge of all derivatives products. Suppose management of an end user firm determines to limit allowable derivatives instruments to Euro futures, Treasury bond futures, and Euro forward contracts. The following attribute-based constraint, named limitProduct, enforces this business rule (Table 9). The IN operator specifies a set of allowable products between a pair of parentheses, and requires the productId attribute to match one of these values.

```
ALTER TABLE AuthorizedProduct ADD CONSTRAINT limitProduct
  CHECK (productId IN
     ("euroFutures", "tBondFutures", "euroForward")
n );
```

Table 9. Attribute-based constraint to limit product types

Internal Controls at the point of Trading

1. Trading only with approved counterparties. Trading only with approved counterparties can be enforced with computerized internal control. If the Counterparty table contains all legitimate counterparties, it can be used to validate trading partners of derivatives transactions. The internal control that any counterparty ID value (i.e. cpId) of the Trade table must also exist in the id column of the Couterparty table can be implemented as a referential integrity constraint in SQL. This referential integrity constraint in turn predicates upon the id column being declared as primary key of the Counterparty table. The primary key constraint, named idAsKey, and the referential integrity constraint, named cpIdAsForeignKey, are as follows (Table 10):

```
ALTER TABLE Counterparty ADD CONSTRAINT idAsKey
   PRIMARY KEY (id);
ALTER TABLE Trade ADD CONSTRAINT cpIdAsForeignKey
   FOREIGN KEY (cpId) REFERENCES Counterparty(id);
```

Table 10. Referential integrity constraint to validate trading partners

The above constraints dictate that every `cpId` value of the `Trade` table be traceable to a valid `id` value of the `Counterparty` table. To illustrate, if the above constraint was in effect, the original attempt to add trade t3990 to the `Trade` table would be rejected by the RDBMS because the named counterparty c10022 cannot be traced to any `id` of the `Counterparty` table. Rejection of invalid data, in turn, helps assure conformity with business rules.

2. Suspending any trader who has exceeded the loss limit. Setting loss limits and requiring traders to abide by these limits is a desirable internal control policy. The `Trader` table, which contains up-to-date gain or loss information, is used to enforce the loss limit. The internal control to suspend traders who have reached or exceeded the loss limits is implemented as the following tuple-based constraint, named `suspendTrader`, in SQL (Table 11):

```
ALTER TABLE Trade ADD CONSTRAINT suspendTrader CHECK
(eId IN
   (SELECT employeeId FROM Trader
     WHERE -1*gainLoss < lossLimit
   )
);
```

Table 11. Tuple-based constraint to enforce loss limit

Whenever changes are made to any tuple of the `Trade` table, the above tuple-based constraint will be activated. This constraint first computes the set of all `employeeId` values from the `Trader` table for traders with losses, if any, below their loss limits. It then tests if the `eId` (i.e. employee ID) value of the tuple in question matches one of the values in this set. The RDBMS enforcing this constraint prohibits any violating record from being saved into the `Trade` table. For instance, as presented in the `Trader` table, the trader with employee ID e633 has exceeded the loss limit. If this constraint was in place, e633 would be barred from making new trades. While this control could not prevent e633's loss from growing to $12 million, it is preventive in the sense that it prevents e633 from

adopting risky strategies such as doubling his bet in an attempt to recoup earlier losses.

Alternatively, an internal control may force the trader with cumulative loss reaching his loss limit to close all positions. This control searches the `Trade` table for all trade IDs associated with the trader's employee ID, and sends corresponding closing orders to the market through a trading system. Additional programming logics in the trading system to sending orders outside of the database are typically coded in a modern programming language such as Java or C++.

3. Checking trader authorization on product type. After determining the authorized product types, the firm needs to check if each trader transacts in only the allowed products. Any trader attempting to trade an unauthorized product should be blocked. This internal control is implemented as a tuple-based constraint, named `checkProduct` (Table 12). If this internal control was in place, the RDMBS would reject the trade t3989 on S&P 500 Futures because trader e633 has no authorization to trade this product.

```
ALTER TABLE Trade ADD CONSTRAINT checkProduct CHECK
(instrument IN
   (SELECT productId FROM AuthorizedProduct WHERE employId=eId)
);
```

Table 12. Tuple-based constraint against unauthorized products

4. Limiting exposure to a given category of product risk. Many derivatives debacles in the past were perpetrated by traders speculating on huge positions which, when the market moves unfavorably, caused massive losses (e.g. JP Morgan in 2012, UBS in 2011, Morgan Stanley in 2008, Societe Generale in 2008, Aracruz in 2008, and Deutsche Bank in 2008). Limiting the size of net positions to an acceptable level is a desirable internal control. For instance, management may decide to limit the notional principal amounts of Euro futures, after offsetting long against short positions, to below $9.5 billion (i.e. $9,500 million). The corresponding internal control is implemented as a tuple-based constraint, named `limitPrincipal`, in SQL as follows (Table 13):

```
ALTER TABLE Trade ADD CONSTRAINT limitPrincipal CHECK
(NOT EXISTS
   (SELECT * FROM
      (SELECT SUM(notionalPrincipal) AS s1 FROM trade WHERE
      instrument='euroFutures' AND position='long') AS T1
      CROSS JOIN
      (SELECT SUM(notionalPrincipal) AS s2 FROM trade WHERE
      instrument='euroFutures' AND position='short') AS T2
      WHERE ABS(s1 - s2) >= 9500
   )
);
```

Table 13: Tuple-based constraint to limit the size of net positions

The `SUM()` function first sums the notional principal amounts of all long Euro futures positions, and assigns this sum value to the `s1` attribute of a temporary single-cell table `T1`. It then repeats the same with all short Euro futures positions, and assigns the sum value to the `s2` attribute of a temporary single-cell table `T2`. Cross joining `T1` and `T2` produces one new row with `s1` and `s2` as the only two columns. The `ABS()` function computes the absolute value of the difference between `s1` and `s2`. The `NOT EXISTS` operator precludes this absolute value from reaching or exceeding 9,500. Accordingly, the RDBMS will reject any new trade that violates this constraint.

5. Limiting exposure to a given category of product risk (for derivatives end users). A number of derivatives scandals involved end users who unnecessarily assumed excessive risk from derivatives transactions (AIJ Investment Advisors 2012, Olympus Corporation in 2011, Aracruz in 2008, CITIC Pacific in 2008, Kashima Oil in 1994, Proctor & Gamble in 1994, Metallgesellschaft in 1993, and Showa Shell Sekiyu in 1993). This and the next internal control are specifically designed to promote prudent use of derivatives by end users.

Unlike investment banks, arbitragers, derivatives brokers/dealers, and speculators who trade derivatives for profits, end users are supposed to use derivatives to hedge. An insurance company, for instance, may maintain a portfolio of Treasury bonds, and use derivatives to hedge the risk of bond valuation losses or interest rate hikes. Since the insurance company keeps a long position on the bonds, its hedging position in Treasury bond futures must be a net short position. An internal control enforcing a net short position on Treasury bond futures is sensible for derivatives end users and other conservative investors. Such

an internal control is implemented in the following tuple-based constraint, named `noLongPrincipal` (Table 14). If this internal control was in effect, it would reject the trade t3990 because this trade would otherwise create a net long position on Treasury bond futures.

```
ALTER TABLE Trade ADD CONSTRAINT noLongPrincipal CHECK
(NOT EXISTS
   (SELECT * FROM
      (SELECT SUM(notionalPrincipal) AS s1 FROM trade WHERE
      instrument='tBondFutures' AND position='long') AS T1
      CROSS JOIN
      (SELECT SUM(notionalPrincipal) AS s2 FROM trade WHERE
      instrument='tBondFutures' AND position='short') AS T2
      WHERE s1 > s2
   )
);
```

Table 14. Tuple-based constraint against net long position

6. Monitoring the daily frequency of trades (for derivatives end users). While investment banks, arbitragers, derivatives brokers/dealers, and speculators engaging in day-trades may reverse their derivatives positions multiple times a day, end users using derivatives for hedging purpose have no need to open and then close a derivatives position within the same day. Systematically prohibiting day trades is a reasonable internal control for derivatives end users and other conservative investors to reduce the risk of reckless speculative trading. The following assertion, named `noDayTrade`, prohibits day-trading of Treasury bond futures (Table 15).

```
CREATE ASSERTION noDayTrade CHECK
(NOT EXISTS
   (SELECT * FROM Trade AS T1, Trade AS T2
     WHERE T1.employeeId = T2.employeeId and
     T1.instrument = 'tBondFutures' and
     T2.instrument = 'tBondFutures' and
     T1.tradeDate = T2.tradeDate and
     T1.position <> T2.position
   )
);
```

Table 15. Assertion against day-trading

Internal Controls Subsequent to Trading

1. Monitoring warning signals on total loss. Many derivatives scandals and debacles culminated in multibillion losses (e.g. Morgan Stanley for $9 billion in

2008, JP Morgan for $8-9 billion in 2012, Societe Generale for $7.2 billion in 2008, Amaranth Advisors for $6.5 billion in 2006, Long Term Capital Management for $4.6 billion in 1998, and Sumitomo Corporation for $2.6 billion in 1996). All internal controls discussed so far, such as suspending traders who have exceeded their loss limits, are *preventive* in nature and represent just a partial defense. Preventive controls can only prevent violating trades, but can do nothing with market-driven valuation changes in existing positions. A useful control over existing positions is to constantly monitor their valuation changes. This and the next internal controls are *detective controls* to monitor unusual losses and gains.

Early warning of unusual losses, before they mushroom to mammoth size, is another desirable strategy to curb the damages. Early loss-warnings can be used to monitor individual traders or the company as a whole. In RDBMS, a trigger can be implemented in SQL to generate exception reports to alert management to unusual changes in trading gains or losses. For instance, when derivatives are actively traded, top management needs to be aware of unusual changes in the total trading gain or loss of all traders. In the trigger below, named `sumOfLossTrigger`, when the total trading loss over all traders exceeds $100 million, a record about the total loss will be added to the `ExceptionReport` table to alert management to the unusual loss situation (Table 16).

```
CREATE TRIGGER sumOfLossTrigger
AFTER UPDATE OF gainLoss ON Trader
@sum_loss = SELECT SUM(gainLoss) FROM Trader;
WHEN (@sum_loss < -100)
   INSERT INTO ExceptionReport
     VALUES (CONCAT
        ('Total loss from derivatives trading is now ',
        @sum_loss, ' million.')
     );
```

Table 16. Triggering exception report on unusual losses

Here, the trigger is activated following every update of the `Trader` table's `gainLoss` attribute. Total gains and losses of all traders are summed by the `SUM()` function, and assigned to the `sum_loss` variable. The trigger then checks if `sum_loss` exceeds $100 million. If it does, a new entry describing the total loss will be inserted into the `ExceptionReport` table. The contents of the

entry is composed by the `CONCAT()` function which concatenates a generic warning message with the value of the `sum_loss` variable.

Top management is supposed to be constantly monitoring the `ExceptionReport` table. When total loss balloons to, say, $103 million, the following warning will be added to the `ExceptionReport` table: "`Total loss from derivatives trading is now -103 million`." Alternatively, database engines integrated with emailing facility can email the warning to responsible members of management.

2. Monitoring the gains of each trader. Derivatives trading is often described as a zero sum game where one trader's gain is another trader's loss. An aggressive trading strategy contributing handsome gains in one quarter could be the culprit for painful losses in another quarter. In fact, some rogue traders amassed huge trading gains early on (Kerviel of Societe Generale in 2007, Brian Hunter of Amaranth Advisors in 2003, Leeson of Barings Bank in 1992, etc.), only to see their aggressive trading strategy inflicting multibillion dollar losses when markets later turned against them. Unfortunately, while huge gains were joyfully celebrated, their risk implications were frequently ignored. For instance, financial officials at P&G, who lost $160 million in 1994 on interest rate swaps, "asked no uncomfortable questions as long as the market was going up and they were making money"(New York Times Apr 14, 1994). In 2008, Kerviel maintained in his defense that "there was a culture of rule breaking at [Societe Generale] where management deliberately turned a blind eye in the hunt for profit" (BBC Oct 5, 2010). In 2012, JP Morgan reported $2 billion of losses, which were later unofficially revised to $8-9 billion, through credit default swaps. The unit trading the swaps earned more than $4 billion in preceding years, and was granted an expanded trading limit. Internal warning about the risk of the gigantic swap position failed to alert top management to the huge potential losses (New York Times May 11 and June 28, 2012).

Ignoring the risk implications of unusual gains is arguably the biggest, saddest, and most tenacious mistake in derivatives internal controls. Management and accountants need to carefully review unusual trading gain incidents and thoroughly understand their risk implications. The internal control to report individual traders' unusual gain incidents, assuming a threshold of $60 million, for

managerial review is implemented as a trigger, named `unusualGainTrigger`, as follows (Table 17):

```
CREATE TRIGGER unusualGainTrigger
AFTER UPDATE OF gainLoss ON Trader
WHEN (NEW.gainLoss > 60)
   INSERT INTO ExceptionReport
   VALUES (CONCAT
      ('Trader ', NEW.employeeID, ' gained ', NEW.gainLoss,
      ' million from derivatives trading.')
   );
```

Table 17. Triggering exception report on unusual gains

This trigger is again activated after every update of the Trader table's `gainLoss` attribute. The `NEW` qualifier refers to the tuple after updating. If the updated gain exceeds $60 million, a new alert message will be inserted into the `ExceptionReport`. To illustrate, if trader e089's gain reaches $62 million, the message to be added to the `ExceptionReport` table will be "`Trader e089 gained 62 million from derivatives trading.`"

7. CONCLUSIONS, LIMITATIONS, AND FUTURE RESEARCH

Learning to monitor and control the risks of derivatives is an important lesson for management and accountants. The current accounting literature seldom investigates internal control in the contexts of computerization, databases, and derivatives. Integrating techniques from accounting and computer science, this is the first paper to illustrate internal control implementation in SQL for derivatives risk management. It is relevant to both the academic and the industry as it contributes to the training of current and future managers and accountants. In particular, it helps business and accounting students understand computerized internal controls for derivatives, and helps professional managers and accountants develop capability in using, assessing, and monitoring such controls.

The first limitation of this study is its simplistic illustrations. For the sake of clarity in presentation, this paper uses simple thresholds and discusses each constraint individually. The reader should expect more complex internal control implementations in real life where multiple constraints may be applied to a single entity. Second, this study represents internal controls in SQL, which however is just one of many query languages or notations for specifying database operations.

Because SQL is executable by database engines and widely used, it is chosen over other representations such as relational algebra, OQL, QUEL, etc. Third, this study focuses on database systems because of their overwhelming popularity, but they are not the only platform where internal controls can be computerized. Many managers and accountants have worked with SQL and databases, but they are less likely to be comfortable with programming languages (e.g. C++, Java, etc.) or XML data representation. Accordingly, this paper stays away from computerized internal controls enabled by programming language codes and XML

While this paper strives to provide simple illustrations for the reader, real life internal controls are more complex. Some of these controls may require intensive processing such as computing the product of multiple tables or executing multiple levels of nested queries. In addition, the complexity of internal controls also grows when more intelligence is to be added. For instance, to use cluster analysis to determine appropriate thresholds, data-mining software need to be integrated with the relational database management systems. With these complexities, the total time taken to execute a computerized internal control to completion may be considerable. Future research on query optimization and on integrating data-mining tools with databases may help create more intelligent and responsive computerized internal controls. Hopefully this paper will be the impetus for more research that integrates accounting and IT concepts and methods to enhance further computerization of accounting systems and internal controls.

8. ACKNOWLEDGMENTS

This paper has benefited from Jeong-Hyon Hwang's inspirational technical advice. The authors are grateful to Rosemond Desir, Hui Du, Christine Earley, Jagdish Gangolly, Guido Geerts, Ram Gopal, John Kiss, Alex Kogan, Gim Seow, Khim Sim, and Dmitry Zhdanov for useful comments. The work of Kinsun Tam was supported by a Faculty Research Award Program grant of the State University of New York at Albany and by the State University of New York at Albany School of Business.

9. REFERENCES

AGRAWAL, R.; JOHNSON, C. M.; KIERNAN, J.; LEYMANN, F. (2006): "Taming Compliance with Sarbanes-Oxley Internal Controls Using Database Technology", *ICDE, IEEE Computer Society*: 92 - 101. http://dx.doi.org/10.1109/ICDE.2006.155

AICPA. (2009): *Content and Skill Specifications for the Uniform CPA Examination*. Ewing, NJ: American Institute of Certified Public Accountants.

BBC. (Oct 5, 2010): *Societe Generale trader Kerviel jailed for three years*. http://www.bbc.co.uk/news/business-11474077

CHEN, P. (1976): "The entity-relationship model: Toward a unified view of data", *ACM TODS*, vol. 1, n. 1: 9-36. http://dx.doi.org/10.1145/320434.320440

CICA. (2007): *Application of Computer-assisted Audit Techniques - Second Edition*. Toronto, ON: Canadian Institute of Chartered Accountants.

CLARK, N.; JOLLY, A. (Jan 25, 2008): *French Bank Says Rogue Trader Lost $7 Billion*. New York Times. http://nyti.ms/Pv6Vyu

COSO. (1996): *Internal Control Issues in Derivatives Usage*. New York, NY: Committee of Sponsoring Organizations of the Treadway Commission, AICPA.

CRAIG, S.; PROTESS, B.; SALTMARSH, M. (Sep 16, 2011): *UBS Faces Questions on Oversight After a Trader Lost $2 Billion*. New York Times. http://nyti.ms/mYZ5V6

GAUTHIER-VILLARS, D.; MOLLENKAMP, C.; MACDONALD, A. (Jan 25, 2008): *French Bank Rocked by Rogue Trader*. The Wall Street Journal. http://on.wsj.com/O3XgKJ

IFAC. (2010): *Handbook of International Education Pronouncements 2010 Edition*. New York, New York: International Federation of Accountants.

IIA. (2009): *Practice Guide/Global Technology Audit Guide 13: Fraud Prevention and Detection in an Automated World*. Altamonte Springs, FL: Institute of Internal Auditors.

ISACA. (2012): *ISACA Model Curriculum for IS Audit and Control, third Edition*. Rolling Meadows, IL: Information Systems Audit and Control Association.

ISDA. (2009): *Over 94% of the World's Largest Companies Use Derivatives to Help Manage Their Risks.* New York, NY: International Swaps and Derivatives Association.

KING, L. (Oct 6, 2011): *UBS: Our risk systems did detect £1.3bn rogue trader*. Computerworld UK. http://bit.ly/n0WZdT

KRISHNAN, R.; PETERS, J.; PADMAN, R.; KAPLAN, D. (2005): "On data reliability assessment in accounting information systems", *Information Systems Research*, vol. 16, n 3: 307-326. http://dx.doi.org/10.1287/isre.1050.0063

MALKIN, L. (Apr 14, 1994): *Procter & Gamble's Tale of Derivatives Woe.* New York Times. http://nyti.ms/fQfuRa

NEW YORK TIMES. (Oct 5, 2010): *Jerome Kerviel*. http://nyti.ms/bO34rx

PROTESS, B.; SORKIN, A.; SCOTT, M.; POPPER, N. (May 11, 2012): *In JPMorgan Chase Trading Bet, Its Confidence Yields to Loss*. New York Times. http://nyti.ms/J4p8yo

PROTESS, B.; WERDIGIER, J. (Sep 15, 2011): *A UBS Trader, From Typical To Rogue*. New York Times. http://nyti.ms/PslYo7

RAYNER, G.; ALLEN, P. (Jan 25, 2008): *Profile: Rogue trader Jerome Kerviel*. The Telegraph. http://bit.ly/1vhKo2

SILVER-GREENBERG, J.; CRAIG, S. (Jun 28, 2012): *JPMorgan Trading Loss May Reach $9 Billion.* New York Times. http://nyti.ms/MsaXFw

THIPRUNGSRI, S; VASARHELYI, M. A. (2011): "Cluster Analysis for Anomaly Detection in Accounting Data: An Audit Approach", *International Journal of Digital Accounting Research*, vol .11: 69–84. http://dx.doi.org/10.4192/1577-8517-v11_4

ULLMAN, J.; WIDOM. J. (2007): *A First Course in Database Systems, third edition.* Upper Saddle River, NJ: Prentice Hall.