QUANTUM MECHANICS AS A BRANCH OF MEREOLOGY

Extended Abstract for PhysComp'96[1]

Tom Etter
112 Blackburn Ave.
Menlo Park, CA 94025

1. Preface

"Mereology? What on Earth is *that*?" you ask. Well you may ask, since you won't find the word in the American Heritage dictionary, nor even in the 8-volume Encyclopedia of Philosophy. However, if you look in the more enlightened Cambridge Dictionary of Philosophy, you'll discover that mereology is "the mathematical theory of parts and wholes." A very useful word, wouldn't you say? How have we managed all these years to get along without it?

In this paper I shall argue that the core of quantum mechanics, by which I mean that part of it that is given by the quantum unitary dynamical law (generalized Schrodinger equation), the Born probability law, and the projection postulate, really belongs to mereology. More specifically, it belongs to the mereology of relations. Looking at quantum mechanics in this way sorts out many things that are very confusing when we try to force it into more traditional molds. Indeed, it entirely gets rid of one great source of confusion, namely "probability waves," simply by rendering them redundant, just as Newton's theory of gravity renders the epicycles of Ptolemaic astronomy redundant. It also finally makes sense of quantum measurement, which has always been a great muddle.

Assuming that this approach works out and really is the way to go, does it mean we must regard physics as at bottom just mathematics? Wouldn't that be a regression to the grandiose a-priorism of an earlier age? At any rate, it would certainly seem to go against the grain of empiricism. Yet a little reflection shows that this is not really so. Let's take a simpler case:

I trust that the reader will agree that 3 and 4 make 7. I'm not now speaking of pure mathematics, but of facts like 3 cows and 4 cows make 7 cows; 3 miles and 4 miles make 7 miles; 3 volts of noise and 4 volts of noise make 7 — wait a minute — make what? Uh, 5 volts of noise. Five??! Oh yes, of course; that's because noise adds in quadrature.

We see from this humble example that mathematics, when you actually apply it to the world we experience, is as empirical, as liable to error, as "falsifiable," as any so-called

[1] Presented at the Fourth Workshop on Physics and Computation, Boston University, 22-24 Nov. 1996, and published in: T. Toffoli, M. Biafore, and J. Leao (eds.), PhysComp96, New England Complex Systems Institute (1996), 110-116. Also on-line in the InterJournal, see www.interjournal.org.

empirical theory. Indeed, when a really great empirical theory like Euclidean geometry or Newtonian mechanics gets refuted, it goes straight to Platonic heaven and becomes mathematics!

I believe that quantum mechanics is headed for this same glorious demise. In the case of quantum mechanics, however, the axioms from which we can deduce its Platonic soul have already made their journey to Platonic heaven, since they are essentially just the axioms of finite set theory. One benefit of this transmigration is that it reveals quantum mechanics to be of a piece with every day causal thinking, which at last makes possible a coherent account of quantum measurement, as mentioned above. But another and more important benefit is something quite new and unexpected, which is the insight that quantum processes and "classical"/causal processes are actually just two small islands in a vast sea of other kinds of processes. Do these other processes actually occur in nature? We don't know, since before now we have not had the mathematics to even imagine them.

2. Relational Databases

"Principia Mathematica" by Russell and Whitehead is one of the most famous books of the century, and one of the least read. In his autobiography Russell complains that only four people in the world had ever read his favorite part of the book, which is the part on the theory of relations. Though I am sure that this select company is larger today, it's still not very large. Fortunately for us, the relational ideas from Principia that we really need have found a happy home in the much more accessible language of relational databases.

What I shall present here parallels a much longer paper [Etter, 1996] which I wrote in the language of probability because I assumed that scientists are more at home with probability theory than with mathematical logic. But that was before I encountered relational databases. This happened serendipitously when the little journal I edit acquired Microsoft Access, in the hope of keeping better track of its subscribers. Browsing through the instruction book, I came across the word 'linking,' a crucial word in my long paper, and to my amazement found it had exactly the same meaning as I had given it there! I quickly realized that this harmony extended to many other relational concepts, and that with only a few strokes I could also define the concepts I was using from probability theory. To top it all off, database language is practically the mother tongue of today's young programmers, the fastest growing population on Earth! Perhaps mine is the naïve enthusiasm of the new convert, but suddenly databases became IT!

Here let me first briefly introduce database concepts in a context where they already keep company with physics, namely in recording the results of an experiment. Consider an experiment to test EPR, like that of Aspect. The outcome of this experiment is fed directly into a computer, where it is registered as a series of records in a database table T. T has five fields: Angle1, Outcome1, Angle2, Outcome2, and HiddenVar. Actually, the computer only fills in the first four fields — God fills in the fifth, and indeed, only God can see the fifth, but we will sometimes pretend we can look over His shoulder. The computer also runs the experiment, choosing the angles for each trial from among the three possibilities 0, 30 and 60 degrees. The outcome fields are binary, with the two spin values Up and Down.

Physically what is happening is that an accelerator is sending out pairs of particles in the so-called singlet spin state that are detected almost simultaneously at remote locations by devices that measure their up-down spins at one of the three angles mentioned above. However, none of the details of this are of any significance for our present purposes except the near-simultaneity of the detections, by which is meant that the time difference between them is too small to allow a light signal to travel from either remote location to the other. Apart from this one physical fact, which shows that the choice of angle at one location can't affect the outcome at the other, we shall focus entirely on the pattern of data in the database table.

To understand the "paradox" of EPR and the force of Bell's theorem, we need first of all to put aside all of our common-sense ideas about causality, and describe the situation very carefully in terms of five key ideas from probability theory, namely sample space, random variable, probability, independence, and conditional independence. Here are these ideas translated into database language:

The sample space is the set of records, e.g., the set of rows in the table on which are recorded the successive trials. Probability will be defined simply as relative frequency in this so-called recordset; exactly how this notion of probability relates to likelihood won't concern us here. Usually probability will be applied to queries. Thus the probability p(Q) of query Q is defined as the proportion of records selected by Q, while the conditional probability p(Q|P) of Q given P is defined as the proportion of those records selected by P that are also selected by Q. A random variable is defined in probability theory as a function on the sample space. In our table, we'll take the fields (columns) to be our random variables, i.e., a field F, considered as a random variable, is the function that takes each record into the value of F for that record. For each value v of F, we can create a query Q that selects just those records having v, and the set of probabilities of such Q's is defined as the probability distribution on F.

Queries P and Q are called independent if p(P&Q) = p(P)p(Q). P and Q are called conditionally independent, as conditioned by a third query C, if they are independent in the recordset selected by C, i.e. if p((P&Q)|C) = p(P|C)p(Q|C). Conditional independence is a very important concept in relational quantum mechanics, and it's important to be aware that it is equivalent to p(Q|(P&C)) = p(Q|C), better known as the Markov property, and also to p(P&C&Q)p(C) = p(P&C)p(C&Q), which I have called separability (for further details and proofs see Etter 1996, Chapter 3.)

A commonsense analysis of this experimental situation leads us to conclude two things: First, since there's no way to signal location 2 with a message about the choice of angle at location 1, the field Outcome2 will be independent of the field Angle1, and similarly Outcome1 and Angle2. And second, the field HiddenVar, which is the supposedly “real” state of the particle pair, both separates Outcome2 from Outcome1 (separation, in the form of conditional independence, is normally interpreted by probability theorists as common cause) and is independent of Angle1 and Angle2. Quantum mechanics, on the other hand, has some definite things to say about the correlation between the fields Outcome1 and Outcome2 as a function of the angles, namely that if the difference between angles is 0 degrees then the correlation is 100%, if 30 degrees then 75% and if 60 degrees then 25%.

Now there is a piece of simple arithmetic called Bell’s theorem that shows that this particular item of common sense and this particular item of quantum mechanics are in blatant contradiction! There is no set of records in a table that can satisfy them both -- one or the other must give.

What does mereology have to say about that? For a starter, it says let’s retreat to operationalism, always a safe hiding place. The table, minus God’s field, is the objective truth, or as close as we can come to it. Whatever created the order in that table is something else, for the moment. What else? Mereology, while it’s our companion in our foxhole, evades that question entirely and says that the best we can hope for by way of an explanation is to represent our table as a composition of simple parts satisfying some very general law. Since tables are relations, we will interpret this to mean lawfully taking apart the mysterious 4-term relation that we observe among angles and outcomes into simpler and more commonplace relations that characterize the “hidden” parts. Mereology does have a more adventurous side, though. Eventually there would come a time to venture beyond the confines of operationalism and put God’s field back in the table, this time as an act of free imagination rather than classical faith. But that’s getting ahead of our story.

3. The mereology of relations

In what sense are tables relations? And what does it mean to take a relation apart?

Consider the relation “nephew." Let’s make a table of all instances of this relation in the town of Palo Alto. It would have two fields, first a field of type PERSON called ‘AuntOrUncle,' and second a field of type PERSON called ‘Nephew’. Its records would represent just those ordered pairs of people in Palo Alto of which the second is the nephew of the first. The technical term for this recordset is the *extension* of the nephew relation in the *set* of residents of Palo Alto. Russell-Whitehead logic is extensional, which means that relations are identified with their extensions. Database logic actually does better than this by bringing in the quite subtle intentional notion of *type*, but such subtleties won't concern us here, so we’ll treat tables simply as relational extensions.

Now a nephew is a son of a sibling. This shows that there is some sense in which we can take apart ‘nephew of’ into two relations, ‘son of’ and ‘sibling of’. In predicate calculus terms, we say that A is a nephew of B iff there exists a person X such that A is a son of X and X is a sibling of B. We can represent this compound character of 'nephew' by adding a third field called 'Sibling' to our table; the records of the new table represent the ordered triples of people in Palo Alto such that the second is a sibling of the first and the third is a son of the second. We abstract the relation 'nephew' from this new table simply by ignoring the Sibling field.

This operation of hiding the sibling data will give us back our original nephew table provided the sibling in each case of the nephew relation is unique. This is ordinarily what we would expect, but it need not always be true, since it's biologically possible for one's nephew to be the son of both one's brother and one's sister. Such "double nephews" would show up twice in the three-record table, once with their mothers in the sibling field and once with their fathers. Dropping the Sibling field would then produce a table differing from the original in that some of the records would occur twice.

Since a double nephew is a pretty special kind of relative, the Palo Alto Bureau of Social Statistics decided it would like to distinguish them from ordinary single nephews, so they compiled a table with a Sibling field. But then they realized that they didn't want to inadvertently stigmatize the double nephews' incestuous parents by putting them on record as such, so they erased the Sibling field and replaced it by a field called 'Count' that contains not the siblings' names but only their number, in this case1 or 2.

It just so happens that there is a standard SQL query called 'Group By' that performs exactly this operation (SQL is the macro-language by which one "questions" the data in programs like Access.) That is, this query hides a specified set of fields, adds a new field called 'Count' which, for each record, is the number of records matching it in the unhidden fields, and then gets rid of all the duplicates. Let's call tables produced in this way *Count tables*. Count tables not only show us the structure of relations in themselves, but also tell us something important about the more complicated background relations from which these "foreground" relations are abstracted. It turns out that this makes them the just right objects of study for a mereology of relations that bears on physics. We'll turn to physics in Section 3 but in this section we'll remain pure mereologists. Here, to start us off, is a very important rule of thumb:

THE GOLDEN RULE OF MEREOLOGY. Always ask the following two questions together:
Q1.) How do you make the whole out of the parts?
Q2.) How do you make the parts out of the whole?

These are two sides of the same question.

Let's ask these two questions together about 'nephew of'. We already have an idea of what the parts of 'nephew of' are, namely 'sibling of' and 'son of', so we'll start with Q1: How do you make the nephews out of siblings and sons?

Suppose we compile two tables, a sibling pair table and a parent-son table. Clearly these two tables together contain all the information we need to construct the three-field table above, and from that the nephew table. But just how do we create this three-field table? It turns out that once again SQL has exactly the operation we need, which is called *inner join*, or simply *join* (there are other kinds of joins, but we won't be concerned with them here.) This operation is a crucial one for "database physics", so let's examine it carefully.

Let's call the fields of the sibling table 'Sibling1' and 'Sibling2'. The sibling relation is symmetrical, so if the ordered pair (Joe Smith, Mary Smith) is a record, then so is (Mary Smith, Joe Smith). The fields of the son table will be 'Parent' and 'Son', which of course is not symmetrical. Now to form the three-field table above, we take three steps:

First we decree that one of the sibling fields, say Sibling2, is going to be equal to the Parent field; this is called *linking* the two fields. Now not every parent is a sibling and not every sibling is a parent, so linking discards records from both tables, namely those from the sibling table for which Sibling2 is childless, and those from the son table for which Parent is an only child.

Next we create a table T with the four fields Sibling1, Sibling2, Parent and Son whose records are all the possible combinations of the values of these four fields that are consistent both with the two tables separately and with the link. To put it another way, a record of T is any sequence of four field values such that the first two are a record in the sibling table, the second two a record in the son table, and the second and third values are equal.

Finally, we hide what is now the redundant field Parent. The resulting three-field table is the join. If we change the name 'Sibling1' to 'AuntOrUncle', 'Sibling2' to 'Sibling' and 'Son' to 'Nephew' to reflect their newly-revealed relational status, we obtain the three-field table described above. If we also hide 'Sibling', the resulting two-field nephew table, with duplicate records removed, represents the Russell-Whitehead *composition* of the two *component* relations 'sibling of' and 'son of'.

The general rules for joining tables T1 and T2 are these: First link a subset (which can be null) of the fields of T1 to a subset of the fields of T2. Next form a table containing all records in the Cartesian product of T1 and T2 that are consistent with the link. Finally, discard redundant linked fields. One can optionally rename other fields to clarify their new status, but in the present theory this is only a "cosmetic" change.

I've gone into a lot of detail about joining since it's the basic concept of the whole theory. Needless to say, we'll have to move on in a much more summary fashion.

The concept of joining can be extended to Count tables in the obvious way: First you make visible those hidden fields whose absence produced duplicate records, then you join the resulting tables as above, then you apply 'Group By' again to hide the hidden fields and supply the appropriate counts. It's easy to see that when you do this, the count in each joint record is the product of its counts in its two separated parts. This product rule is essentially what leads to Hilbert space and quantum mechanics.

It turns out that it's sometimes useful to add records with counts of 0 to a table. In a sense, these "null" records signify nothing; the 0 says, in effect, "I'm not here". Still, these 0 counts are consistent with the product rule for joins, which makes them useful in the algebra, like the null set in set theory, and they are otherwise harmless.

Much more serious, though, is the need to sometimes count records *negatively*! Like 0 counts, these don't seem to make much difference for the basic mathematics - indeed they simplify it, just as negative "dollars" simplify accounting. Conceptually and philosophically though, negative cases, which discretely tiptoed into science about 70 years ago with the canceling "amplitudes" in the quantum two-slit experiments, are probably the most subversive novelty in science since 1600. Indeed, I'll wager that before too long they are going to profoundly transform the very nature of science as a human enterprise, and scientists in a few hundred years will scratch their heads in utter amazement at our superstitious ravings today. Such speculation is out of bounds in this paper, though, which will try to remain cheerfully in step with the march of progress as we currently understand it.

Back to the golden rule: What about Q2, the other side of the part-whole question? How do we make the parts out of the whole?

How do we make siblings and sons out of nephews? The simple answer is that we don't. There's a more complicated answer that we'll come to shortly. But for now let's take an easier question: How do we make the sibling table and the son table out of the three-field table that has a sibling field? This one has an easy answer: Make the sibling table simply by hiding the Son field and removing duplicate records, and make the son table simply by hiding the AuntOrUncle field and removing duplicate records.

Is this a general answer? No! Given a Count table T with fields F1, F2 and F3, there does not in most cases exist a Count table T1 with fields F1, F2 and a Count table T2 with fields F2, F3 whose join is T. The necessary and sufficient condition for T1 and T2 to exist is that F2 separates F1 from F3 in the probabilistic sense defined in Section 2. This theorem, which is proved in [Etter 1996], shows how intimate is the connection between probability and logic!

Even when the separability structure of a complex table allows us to take it apart, its parts are in general not unique. Furthermore, seemingly arbitrary decisions about what these parts should be may affect their internal separability structure in various ways, and thus bear on our ability to break them into smaller parts. The situation is really quite complicated. In general the parts of a table are not simply "there" in the table, nor are they simply supplied by our imagination. We are free to choose many aspects of the parts, but these choices have consequences, and some choices are much better than others. It may help here to remind us of the similar complexities and ambiguities involved in choosing the “right” coordinate system.

We started with Q1 plus a certain notion of what the parts were. But suppose we had started out cold with Q2; the problem then is not just to find the parts but also to define what we even mean by part. If we didn’t know about joining, it’s very unlikely we would have thought of the parts of a table as the component tables defined above, and we probably would have taken the table apart into more obvious parts like records or fields. Indeed, why not? Let’s try out fields as table parts.

This brings up a crucial distinction: that between a *map* and a *diagram*. A table is essentially a *map* of a relation, and fields are *regions* of that map, which is to say, they are *parts in place*. Notice that the order of records in a table is immaterial. Changing that order, which happens all the time in the normal operation of a database program, doesn’t change the relation that the map represents; it gives a different map of the same territory, so-to-speak. Thus if we are given a field in isolation, we can have no idea what relation that field partially represents. In effect, an isolated field is simply a heap of values - let’s call it a *value-set*. Applying the golden rule, we must now ask: how do we put such value-sets back together into a table?

It’s not easy. It’s a bit like putting hamburger back together into a cow. It requires that we know for each value set just how it lines up with all the others, e.g., it requires that we re-supply *all* of the relational information that was in the table! In this case, the whole is indeed more than the sum of its parts, and such is generally true for the parts of a map.

On the other hand, the set of *components* of a table contains almost all of the table’s relational data. That’s because we can uniquely define the whole table, modulo the order of its records, simply by drawing a “wiring diagram” of these components showing which fields are linked to which. Such a diagram is quite close to being a “sum of parts”; it’s that plus the small amount of extra information needed to specify which fields are linked to which. Note that in most cases the table of a link diagram has exponentially more records in it than are in all of its components taken together (the exponent being the number of components), so taking apart a table into linked tables can enormously reduce its redundancy, a point that has not been lost on database designers. As a rule, natural science makes maps, while engineering and theoretical science make diagrams.

Finally, let’s return to the hard question: How do we find the components of a two-field nephew table? We cannot just cut the table in two as it stands; rather, we must somehow supply a new field as the cutting place. Lacking any other data, we have no choice but to construct this field as “an act of free imagination," as Einstein once described theory making. Of course we often do have other data. Given a two-field input-output table for a computer, for instance, we can find many of the hidden fields by taking the cover off the computer and probing with an oscilloscope. In the good old days of discrete components, we could even take these hidden fields apart with wire-cutters and soldering iron, leaving us with a heap of simple truth tables, so to speak.

This experience of taking apart and putting back together our mechanical artifacts leads us to naively imagine a similar possibility for places like the quantum realm. It's as if we had spent our lives in a house with a checkerboard tile floor, and naively project the existence of such tiles to the whole Earth's surface, including water tiles on the ocean. Behind this obvious mistake there is a subtle truth, however. Water tiles don't actually exist, but it turns out that by *imagining* that they do, we can construct a mathematical form called a Reimanian coordinate system that is immensely helpful in describing the ocean surface and how it changes. Let this example give us the courage, or maybe we should call it the presumption, to start imagining little database tables in the mysterious and watery realm of the quantum.

4. Von Neumann states and linear dynamics

Von Neumann defined a quantum state as a property of an *ensemble* of quantum objects. Let's imagine this ensemble to be a series of particles emitted by some kind of laboratory device, and suppose we can test or *query* the emerging particles for various properties. Each such query Q has a probability *p(*Q*)* defined as the proportion of particles that pass its test. The set of all these probabilities is what von Neumann called the *state* of the ensemble. If *p(*Q*) =1*, then it's safe to say that Q is a property of each individual particle; otherwise it may not be clear whether we should regard Q is a property of the individuals, or only as a property of the group. If every particle that passes a query R will subsequently pass a query Q, but not vice-versa, we say that R *refines* Q. If there is a Q such that *p(*Q*)=1* that has no refinement, we say that the state is *pure*, otherwise *mixed*.

Von Neumann found a very elegant way to characterize a quantum state, pure or mixed, as a matrix of complex numbers that he called the *density matrix*. For a pure state, the density matrix is the outer product of the wave function vector and its dual, while for a mixed state it is a linear combination of orthogonal pure matrices weighted by their probabilities. The way he connects these matrices to the ensemble probabilities is beautifully simple. First of all, he shows that the physically meaningful queries are in natural 1-1 correspondence with so-called *projection* matrices. To see what he is doing here, first note that a projection matrix is a matrix for which there is a basis in which it has 0's and 1's in the diagonal and 0's elsewhere, and also note that if several pure states pass a certain query, so do all their linear combinations. The matrix associated with a query Q is that such that its 1's define a basis for the subspace of pure states that always pass Q, and its 0's define a basis for the subspace of those that always flunk; call this matrix *Q*. This association leads to the following general rule for *p(*Q*)* given a state matrix *S*: *p(*Q*) = trace(QS).* For pure states this rule is equivalent to Born's rule, and for all states it is deducible from Born's rule. It can be shown that no matrix besides S will give the same *p(*Q*)* for every Q, so we see that the *S*'s uniquely represent the states as originally defined.

The state matrix evolves according to another simple law, $S' = TST^{-1}$, where T is a unitary matrix representing the passage of time. Note, and this will be important, if we multiply both sides on the left by T we get the law $S'T = TS$, which is more general in the sense that it doesn't require that T have an inverse. From this dynamical law plus the above probability law we can reconstruct the entire basic machinery of operators that is usually formulated in terms of wave functions. Von Neumann introduced density matrices using rather complicated and arcane mathematics, but others have simplified his account (see Mackey, 1963 and Jauch, 1968). Still, the question remains, what does it all mean? Why should a state be a matrix? Why does time produce a linear transformation of that matrix? Why should probability be the trace of a matrix product? So long as one stays in the murky shadow of wave mechanics, and especially so long as one keeps trying to make neo-Newtonian models of "probability waves", these questions have no intelligible answers. But don't despair - help is on the way. Enter the mereology of relations!

Define a *closed diagram* as a diagram in which every field is linked to another field. We say that two diagrams T1 and T2 are *equivalent* if the join of T1 is the same table as the join of T2. There's a way to transform any diagram into an equivalent closed diagram, which is to link all its single fields to *Zilch tables*, defined as single-field Count tables in which every field value occurs just once. When we join the components of a closed diagram, the linked pairs in the diagram turn into the individual fields of the resulting table, so it makes sense in such a *closed* table to speak of the *link* of any field.

Let F be a field of table T — we are now speaking of Count tables. If we hide all the other fields, we get a new Count table V with the single field F (not counting the Count field) whose counts define the probability distribution on F. We can think of V as a vector indexed by the values of F; call this the *state vector* of F. Doing the same with a pair of fields F and G leads to a joint *count matrix* indexed by the values of F and G that defines the joint probability distribution on F and G. Now we are finally in a position to define the key concept that ties together quantum mechanics and the theory of relations, which is the concept of *density matrix*. Informally, the density matrix of F is the count matrix you get when you cut the link of F. More formally:

DENSITY MATRIX. Let T be a Count table, let F be a field of T, let D be a closed diagram of T, let F' be the field linked to F in D, and let D' be the diagram that results from removing that link. The *density matrix* of F is then the count matrix on F and F'. We'll also refer to this matrix as the *state matrix* of F, or, when we don't need to distinguish it from the state vector, as simply the *state* of F.

It's easy to see that the state vector V is the diagonal of the state matrix S. Let Q be a query about F; we can represent the selection made by Q as a characteristic function on the values of F, i.e. a function that is 1 for values in the selection, 0 for values that are not. Let's place this function in the diagonal of a square matrix indexed by the values of F; the result is a projection matrix that we'll call Q. It then follows immediately that $p(\mathrm{Q})$

= *trace(QS)/N*, where *N* is the total count. In other words, our state matrices, when we normalize them by the total "case count," satisfy the von Neumann probability rule!

Suppose T is a two-field table in a diagram. The count matrix of T with its links cut will be called a *transformation*, designated by *T*. The question now is: what is the relationship between the density matrices *S* and *S'* of the two fields of T? Answer: *S'T* = *TS*. In other words, the state matrices of any two-field component of a table satisfy the von Neumann dynamical rule!

When you link two-field tables, you multiply their matrices. Now with negative counts allowed, it may well happen that a table matrix *T'* is the inverse of a table matrix *T* (ignoring normalization), so when you link these tables and hide the linked field you get the "identity" table. In particular, you can always use this construction to take apart a link itself. Suppose we do this to a link whose state is *S*, and then apply a query Q to the "hidden" field linking T and T'. Then $p(Q) = trace(STQT^{-1}) = trace(SQ')$, where *Q'* is a non-diagonal projection matrix. If we do this for all unitary T's, we obtain precisely the set of queries that von Neumann used to define the quantum state of an ensemble in terms of measurement.

Though these results are suggestive, they don't give us quantum mechanics proper. Our next step is to carefully distinguish among several kinds of state matrices. First, let's remind ourselves of just how general our present conception of state actually is. Database tables can represent an immense range of different kinds of things — mailing lists, actuarial tables, a logic wiring diagram, computer programs, systems of equations, indeed anything at all that involves organized data. A table is simply a relationship among data. The counts in a Count table point beyond the data itself to how it emerges from, and bears upon, an otherwise invisible background, but this counting in no way limits what the table represents. State matrices arise from the bare act of taking a Count table apart, so it is not surprising that there are states of many kinds other than quantum. There are only a few formal classes of states, though, of which quantum is one.

A distinction that applies to states of all kind is that between pure and mixed states. A state is *pure* if its field separates its table into two parts. Any other state is mixed. Pure states are easier to think about than mixed states, but there's no way to avoid the latter if there are any irreducible cycles in the diagram. A pure state is always the outer product of two vectors; in Dirac notation we have $S = |v\rangle\langle w|$.

If a diagram is a cycle-free functional composition, with or without "random inputs," then all of its states are pure and of the form $|v\rangle\langle Z|$, where $\langle Z|$ is the "white" vector of a Zilch table. I have called such states *causal*. "Statistico-causal" or "Markovian" would be more literally accurate, but these are too awkward for a concept that has such a key role in our experience of process. Since linking to a Zilch table does nothing, a causal state is completely characterized by the vector $|v\rangle$, which is in fact the state vector *V* of its field. The notion of link state doesn't really belong to causal thinking at all. Mixed causal link states can be used to neatly describe recursion, but underlying the loops of recursion there is always a cycle-free causal substratum. Science has never before clearly recognized any notion of process that is not causal in the above sense, and much of the confusion in the philosophy of quantum mechanics has come from trying to force quantum states into the causal mold.

A pure quantum state is characterized by the property that its constituent vectors are equal (this is not quite formally true for complex quantum mechanics, but as Mackey shows, complex quantum mechanics is shorthand for real quantum mechanics subject to certain symmetry conditions; see Etter, 1996). The vector v in the state $|v><v|$ is the famous, or infamous, probability wave of standard theory. Notice that v is not a feature of the quantum table T itself, but of an imaginary table that results from "cutting a wire" in a diagram of T. Since there is no more hope of physically cutting this wire than there is of physically prying loose a "water tile" out of our Riemannian coordinate system, we must treat both tile and wire as "free constructions of the imagination".

Notice also that the ``amplitudes'‘ of this wave v are (normalized) record counts, which is to say, they are probabilities in exactly the sense that the values of the table state vector V are probabilities. The square law arises simply from the fact that the diagonal counts in $|v><v|$ are the squares of the counts in v. There's no room at all here for deterministic hidden variables, since variables are actual, whereas v is a property of the imagined cut ends of the link, which hasn't actually been cut. The wave function is a useful enough concept in its place, but it *doesn't exist*!

The more general condition defining a quantum state is that it is unchanged by reversing rows and columns, and that it has no negative counts in its diagonal. The theory of tables in which all states are quantum, and all transformations are unitary, is equivalent to Hilbert space quantum mechanics, or more exactly, approaches it in the limit as we let the tables have arbitrarily many records. Actually, it gives quantum mechanics minus the von Neumann projection postulate, which can only be proved, indeed which only has meaning, after we embed the quantum table in a larger table that also has causal states representing what happens in the laboratory. Analyzing this larger context has finally led to a coherent account of quantum measurement, which I count as the chief accomplishment of the theory to date [Etter, 1996].

At the end of Section 2, I mentioned that the way to deal with EPR is to analyze the strange data table into simpler and more understandable parts. The above-mentioned relational context that merges quantum and causal enables us to do just that, and it turns out that the quantum part of the diagram is almost trivially simple; the hard work is in carefully formalizing the "classical" aspects of the situation. But that's another paper.

This has been a very quick run through a lot of material, much of which was necessarily over-simplified. You can take the same tour at a more leisurely pace in [Etter, 1996], though in a different language. Let me close on my opening note: Granted that this new mathematics *could* explain quantum mechanics, how do we know that it actually does? Suppose I am looking out the window and see three apples fall out of the tree, followed by four more. A short while later, I go out and find seven apples on the ground. Is this explained by 3+4=7? Not necessarily. It could be that squirrels carried off the first batch and seven more fell while I wasn't looking. The simpler, more reasonable and more fundamental explanation is not always the right one. In the quantum case, however, it's the one I'll lay my bets on.

REFERENCES

Etter, Tom, [1996] "Process, System, Causality and Quantum Mechanics", Forthcoming in *International Journal of General Systems*, special issue on "General Systems and the Emergence of Physical Structure from Informational Theory".

Jauch, Josef M, [1973], *Foundations of Quantum Mechanics*, Addison-Wesley, London.

Mackey, George W., [1963], *Mathematical Foundations of Quantum Mechanics.* W. A. Benjamin, New York.

von Neumann, John, [1955] *Mathematical Foundations of Quantum Mechanics.* Princeton University Press.