(1) A category $\mathcal{C}$ is said to be cartesian iff it admits finite products — in other words, iff it has a terminal object, which we shall write 1 or $\top$, and every two objects A and B have a product which we write $A \times B$ or $A \wedge B$.

(2) A category $\mathcal{C}$ is said to be cartesian closed iff it is cartesian and for every object A, the contravariant functor $— \times A$ which takes an object C to $C \times A$ (and a morphism $f: C \to C'$ to $(f, 1_A): C \times A \to C' \times A$) has a right adjoint. This right adjoint will be written $(—)^A$ or $— \Leftarrow A$ or $A \Rightarrow —$. We call B^A the internal hom of A and B, or the exponential of B by A. It is characterized by the fact that every morphism $C \times A \to B$ determines uniquely and naturally a morphism $C \to B^A$. We also write this

$$\frac{C \wedge A \vdash B}{C \vdash A \Rightarrow B}$$

(when talking of proofs, we shall generally write $\vdash$ instead of $\to$) and we call it the deduction theorem.

(3) Anticipating on what follows, the objects in a cartesian closed category will also be called the types of the category, or its statements. An arrow $f: X_1 \times \cdots \times X_n \to A$ will be called a term of type A with free variables of type $X_1, \ldots, X_n$, or alternatively a proof of A from the hypotheses $X_1, \ldots, X_n$. When $n = 0$, this reduces to $f: 1 \to A$, and we say that f is a global point of A, or a (closed) term of type B, or a proof of A (with no hypothesis). We note that it is always possible (by composing with the appropriate projection arrow) to *add* free variables to a term, and we shall not generally distinguish two terms that differ only by dummy free variables.

(4) The identity morphism $B^A \to B^A$ gives, under the functor adjunction, a morphism $B^A \times A \to B$ (which is none other than the counit of the adjunction) which we call the evaluation map ϵ, which we can also write $(A \Rightarrow B) \wedge A \vdash B$. Thus, if $g: X_1 \times \cdots \times X_n \to B^A$ is a term of type B^A with free variables of types $X_1, \ldots, X_n$ and $f: X_1 \times \cdots \times X_n \to A$ is a term of type A with free variables of types $X_1, \ldots, X_n$, we can form a term $\epsilon(g, f): X_1 \times \cdots \times X_n \to B$ of type B with free variables of types $X_1, \ldots, X_n$, called the composition of g and f (though the term "application" is probably better), and written $g^\int f$, or simply gf when no confusion can occur. We also make the convention that hgf will stand for $(hg)f$ (which is in general distinct from $h(gf)$). To summarize, a term of type B^A and a term of type A can be composed to give a term of type B, provided they have the same free variables (which we can always assume). We write this

$$\frac{\Gamma \vdash A \Rightarrow B\ ,\ \ \Gamma \vdash A}{\Gamma \vdash B}$$

and we call it the rule of *modus ponens*.

(5) On the other hand, if we have $f: X_1 \times X_n \to B$ a term of type B with free variables of types $X_1, \ldots, X_n$, and $A = X_i$ is one of these types, we can use the deduction theorem to form a new term of type B^A with free variables of types $X_1, \ldots, \hat{X}_i, \ldots, X_n$: we shall write it $\lambda[x_i : X_i].f$, and we say that it is obtained from f by abstraction of the i-th free variable. Note that in the process of writing f we shall write x_i for the identity on X_i so that we know what variable we are abstracting.

(6) To take a few examples, if A is a type, we have a term $I_A = \lambda[x : A].x$ of type $A \Rightarrow A$, which we shall (somewhat abusively, of course) call the identity on A (in fact, it is an arrow from $\top$ to A^A, but there is no harm in seeing it also as an arrow from A to A). If A and B are types, we have a term $K_{A,B} = \lambda[x : A].\lambda[y : B].x$ of type $A \Rightarrow (B \Rightarrow A)$. And if A, B, C are types, we have a term

$$S_{A,B,C} = \lambda[x : A \Rightarrow (B \Rightarrow C)].\lambda[y : A \Rightarrow B].\lambda[z : A].xz(yz)$$

of type $(A \Rightarrow (B \Rightarrow C)) \Rightarrow (A \Rightarrow B) \Rightarrow (A \Rightarrow C)$. When writing such terms we shall often refrain from explicitely writing the type of each variable in an abstraction, and leave it to the reader to calculate them. So we might simply write $S_{A,B,C} = \lambda x.\lambda y.\lambda z.xz(yz)$ which we abbreviate as $S_{A,B,C} = \lambda xyz.xz(yz)$.

(7) We note that every term which can be formed from the composition and abstraction operations can actually be formed by composition from its free variables and from the three terms I, K, S. Indeed, to do this we only have to show that we can eliminate one level of abstraction. But $\lambda x.x = I$; and if f does not involve x then $\lambda x.f = Kf$; and for composition we have $\lambda x.fg = S(\lambda x.f)(\lambda x.g)$, so that we can propagate the abstraction downwards until it hits either a variable x alone or a term that does not involve x. For example, we have

$$\begin{aligned}\lambda x.\lambda y.yx &= \lambda x.S(\lambda y.y)(\lambda y.x) = \lambda x.SI(Kx)\\ &= S(\lambda x.SI)(\lambda x.Kx) = S(K(SI))K\end{aligned}$$

(where we have used the fact that $\lambda x.Kx = K$). In fact, I also can be eliminated since $I = SKK$. Of course, we shouldn't be omiting the indices: if x has type A and y has type $A \Rightarrow B$, the expression above has type $A \Rightarrow (A \Rightarrow B) \Rightarrow B$ and should be written

$$S_{A,(A\Rightarrow B)\Rightarrow A,(A\Rightarrow B)\Rightarrow B}(K_{((A\Rightarrow B)\Rightarrow A)\Rightarrow(A\Rightarrow B)\Rightarrow B,A}(S_{A\Rightarrow B,A,B}I_{A\Rightarrow B}))K_{A,A\Rightarrow B}$$

(8) We shall speak of the proofs K and S (of $A \Rightarrow (B \Rightarrow A)$ and $(A \Rightarrow B \Rightarrow C) \Rightarrow (A \Rightarrow B) \Rightarrow A \Rightarrow C$ respectively) as <u>axioms</u>. In section (7) we have shown that every proof that can be obtained from the rule of *modus ponens* and the deduction theorem can actually be obtained with just *modus ponens* and the axioms K and S. In other words, we have "proven" the deduction theorem (provided one admits axioms K and S).

(9) There exists a unique term of type $\top$ with any kind of free variables: we shall write it $\star$, and we also call it an axiom. Similarily, for the $\wedge$ operation, we have three rules of term creation: the two projections $A \wedge B \vdash A$ and $A \wedge B \vdash B$, and the pair creation rule $A, B \vdash A \wedge B$ (this really means $A \wedge B \vdash A \wedge B$, so it is really the identity map on $A \times B$!). These rules allow us to create three important terms which we call axioms: $p_{1,A,B} = \lambda[(x, y) : A \wedge B].x$ (of type $A \wedge B \Rightarrow A$), $p_{2,A,B} = \lambda[(x, y) : A \wedge B].y$ (of type $A \wedge B \Rightarrow B$), and $P_{A,B} = \lambda[x : A].\lambda[y : B].(x, y)$ (of type $A \Rightarrow B \Rightarrow A \wedge B$), with the obvious notations (namely that (x, y) means the term of type $A \times B$ that corresponds to the terms x and y of types A and B respectively by the universal property of $A \times B$).

(10) A cartesian closed category is said to be bicartesian closed iff it admits finite coproducts — in other words iff it has an initial object, which we shall write 0 or $\bot$, and every two objects A and B have a coproduct which we write $A + B$ or $A \vee B$. In a bicartesian closed category, we also write $\neg A$ for $A \Rightarrow \bot$.

(11) A term x of type A defines a term of type $A \vee B$ which we shall write $x^{\leftarrow}$, and similarily for a term of type B which defines a term $x^{\rightarrow}$ of type $A \vee B$. These are two of the three rules for disjunction, namely the two canonical maps $A \vdash A \vee B$ and $B \vdash A \vee B$. They give rise to two axioms $q_1 = \lambda[x : A].x^{\leftarrow}$ which is a proof of $A \Rightarrow A \vee B$, and $q_2 = \lambda[x : B].x^{\rightarrow}$ which is a proof of $B \Rightarrow A \vee B$. To define a term of type $A \vee B \Rightarrow C$, things are a bit more complicated: by the universal property of $A \vee B$ (and one more of a great number of applications of the deduction theorem), such a term corresponds bijectively to a term of type $A \Rightarrow C$ together with a term of type $B \Rightarrow C$. We say that we are defining a term by "discriminating cases", and we shall write $\lambda[x : A \vee B].(f_1|f_2)$ where f_1 is a term in which x appears free with type A and f_2 is a term in which x appears free with type B. Except that to remove the ambiguity as to which variable we are discriminating for, we shall write $x_{\leftarrow}$ instead of x in the term f_1 and $x_{\rightarrow}$ instead of x in the term f_2. This allows us to define another axiom, namely $Q = \lambda[x : A \Rightarrow C].\lambda[y : B \Rightarrow C].\lambda[z : A \vee B].(xz_{\leftarrow}|yz_{\rightarrow})$, which has type $(A \Rightarrow C) \Rightarrow (B \Rightarrow C) \Rightarrow A \vee B \Rightarrow C$. To illustrate some more the rules we propose for writing terms, we give the example of the term $\lambda[x : A \vee B].((x_{\leftarrow})^{\rightarrow}|(x_{\rightarrow})^{\leftarrow})$: this term has type $A \vee B \Rightarrow B \vee A$ and is precisely equal to Qq_2q_1.

(12) Finally, we should say something about $\bot$. First of all, how to define a term of type $\bot$: one just doesn't. Second, how to define a term of type $\bot \Rightarrow A$: the universal property of $\bot$ tells us that there is a unique proof $\bot \vdash A$, and the deduction theorem then lets us manufacture a term of type $\bot \Rightarrow A$, which we write $F = \lambda[x : \bot].()$, and this we call also an axiom.

(13) So let us recapitulate the axioms we have set up:

Name	**Value**	**Type**
K	$\lambda[x : A].\lambda[y : B].x$	$A \Rightarrow B \Rightarrow A$
S	$\lambda[x{:}A{\Rightarrow}B{\Rightarrow}C].\lambda[y{:}A{\Rightarrow}B].\lambda[z{:}A].xz(yz)$	$(A \Rightarrow B \Rightarrow C) \Rightarrow (A \Rightarrow B) \Rightarrow A \Rightarrow C$
$\star$	$\star$	$\top$
p_1	$\lambda[(x, y) : A \wedge B].x$	$A \wedge B \Rightarrow A$
p_2	$\lambda[(x, y) : A \wedge B].y$	$A \wedge B \Rightarrow B$
P	$\lambda[x : A].\lambda[y : B].(x, y)$	$A \Rightarrow B \Rightarrow A \wedge B$
F	$\lambda[x : \bot].()$	$\bot \Rightarrow A$
q_1	$\lambda[x : A].x^{\leftarrow}$	$A \Rightarrow A \vee B$
q_2	$\lambda[x : B].x^{\rightarrow}$	$B \Rightarrow A \vee B$
Q	$\lambda[x{:}A{\Rightarrow}C].\lambda[y{:}B{\Rightarrow}C].\lambda[z{:}A{\vee}B].(xz_{\leftarrow}\|yz_{\rightarrow})$	$(A \Rightarrow C) \Rightarrow (B \Rightarrow C) \Rightarrow A \vee B \Rightarrow C$

(14) These axioms have the interesting property, analogous to that which we saw in section (7) for only one connective, that every term that can be formed with the rules described can be actually written with just the axioms (and the free variables) using composition (that is, *modus ponens*) as the only rule. In particular, any closed term can be written with just the axioms. To give just one example, take the rule of de Morgan:

$$(\lambda[(f,g):(\neg A)\wedge(\neg B)].\lambda[x:A\vee B].(fx_{\leftarrow}|gx_{\rightarrow}),$$
$$\lambda[h:\neg(A\vee B)].(\lambda[x:A].hx^{\leftarrow},\lambda[x:B].hx^{\rightarrow}))$$

of type

$$(((\neg A)\wedge(\neg B))\Rightarrow(\neg(A\vee B)))\quad\wedge\quad((\neg(A\vee B))\Rightarrow((\neg A)\wedge(\neg B)))$$

(where we recall that $\neg A$ means $A\Rightarrow\bot$): it can be written

$$P(S(S(KQ)p_1)p_2)(S(S(KP)(S(S(KS)K)(Kq_1)))(S(S(KS)K)(Kq_2)))$$

(Nobody said it was interesting.) Since we're talking about de Morgan, there is another rule of de Morgan, namely

$$\lambda[h:(\neg A)\vee(\neg B)].\lambda[(x,y):A\wedge B].(h_{\leftarrow}x|h_{\rightarrow}y)$$

of type

$$((\neg A)\vee(\neg B))\Rightarrow(\neg(A\wedge B))$$

which can be written

$$Q(S(S(KS)K)(Kp_1))(S(S(KS)K)(Kp_2))$$

But that is all: there is no mysterious fourth rule, nothing like $(\neg(A\wedge B))\Rightarrow((\neg A)\vee(\neg B))$...

(15) A bicartesian closed category is said to be <u>boolean</u> iff the natural morphism $A\vdash\neg\neg A$ (obtained from the term $\lambda[x:A].\lambda[h:\neg A].hx$ of type $A\Rightarrow\neg\neg A$ by the deduction theorem) is actually a (natural) isomorphism. In that case, we have a term W of type $\neg\neg A\Rightarrow A$, which we call an axiom. For example, we can compose it with the term

$$\lambda[h:\neg(A\vee\neg A)].h(\lambda[x:A].hx^{\leftarrow})^{\rightarrow}=SI(S(Kq_2)(S(S(KS)K)(Kq_1)))$$

of type $\neg\neg(A\vee\neg A)$ to obtain a proof $W(SI(S(Kq_2)(S(S(KS)K)(Kq_1))))$ of $A\vee\neg A$, the so-called *tertium non datur*. We can also form the term

$$\lambda[h:(A\Rightarrow B)\Rightarrow A].W\lambda[k:\neg A].k(h\lambda[\ell:A].F(k\ell))$$

of type $((A\Rightarrow B)\Rightarrow A)\Rightarrow A$, called "Pierce's law".

(16) We recall that any partially ordered set (poset) $\mathcal{P}$ can be made into a category by decreeing that there is exactly one morphism $p\to q$ whenever $p\leq q$, and none otherwise. Now if we translate the conditions for $\mathcal{P}$ to be a bicartesian closed category in terms of the order structure, we find the following: $\mathcal{P}$ is a lattice with a least element $\bot$ and a greatest element $\top$, and for any two elements p,q of $\mathcal{P}$, the set of elements r such that $r\wedge p\leq q$ has a greatest element, written $p\Rightarrow q$: we have $r\leq(p\Rightarrow q)$ iff $z\wedge p\leq q$. Such a poset we call a Heyting algebra. Since $— \wedge p$ is a left adjoint, it preserves coproducts, so we have $(q \vee r) \wedge p = (q \wedge p) \vee (r \wedge p)$ (really, isomorphic, but isomorphic objects in a poset are equal), and the underlying lattice to a Heyting algebra is therefore distributive. In a Heyting algebra, we let (according to the general definitions given earlier) $\neg p$ stand for $p \Rightarrow \bot$: in other words, $\neg p$ is the greatest of those elements q such that $p \wedge q = \bot$. We then have $p \wedge \neg p = \bot$, but it does not follow that $p \vee \neg p = \top$, only that no q with $p \wedge q = \bot$ will make $p \vee q$ any bigger. Any boolean algebra, of course, is a Heyting algebra (with $p \Rightarrow q = q \vee \neg p$), but the converse is not true: the lattice of open sets on any topological space (or indeed of any locale) is a Heyting algebra, and it is a boolean algebra iff the space is a direct sum of indiscrete spaces. Actually, a Heyting algebra is boolean iff it is boolean as a bicartesian closed category (as defined in (15)).

(17) We now proceed to introduce quantifiers in our system. Before we do so, we digress a little from the path of pure rigour and explain something of what we are trying to achieve. We are trying to define something like $\forall(x : U).V_x$ and $\exists(x : U).V_x$, which should be terms, that is, objects of our category. In this construction, V_x should be a term which depends on the variable x, and we take respectively the product and the coproduct of the corresponding terms, in accordance with the philosophy that in intuitionist logic, a proof of $\forall(x : U).V_x$ is a function that takes every x to a proof of V_x and a proof of $\exists(x : U).V_x$ is obtained by giving an x together with a proof of V_x. However, things are not altogether clear, because it is not clear what U is, and even less what x is, or how we should describe an type that varies in function of a variable: if we start writing things like $\lambda[x : U].V_x$, this looks suspiciously like a term, so it should have a type. But on the other hand, if we have a proof $v_x{:}\, \top \vdash V_x$ of V_x for every x (whatever that means!) we get a term $\lambda[x : U].v_x$ of type $\forall(x : U).V_x$, and the relations between the objects $\lambda[x : U].v_x$, $\lambda[x : U].V_x$ and $\forall(x : U).V_x$ are hard to describe in the framework that we have been using so far. Note that here we are forced to introduce one more level: proofs reside on one level, their types (viz.the propositions they prove) on another, and these types seem themselves to be terms of some kind, since it is possible to use some operations like λ on them. Now, U must reside on the highest level, but there is some confusion of the levels because U is quite similar to the type $\tilde{U}$ defined as $\exists(x : U).\top$. For example, if V does not depend on x, then $\exists(x : U).V$ is none other than $\tilde{U} \wedge V$, and similarily, $\forall(x : U).V$ is none other than $\tilde{U} \Rightarrow V$.

Now that we have described the difficulties, how do we solve them? I believe the key to the problem lies in the complex hierarchy we mentioned. For example, take the case of Peano arithmetic: we can quantify over the natural numbers. So we will have to introduce a type, N, for the natural numbers, together with a term O of type N (the number zero, which we write O so as not to confuse it with an initial object), and a successor function S of type N^N. Note that we use the non-logical (but not illogical!) notation here because it doesn't make much sense to speak of S as a proof of the fact that N implies N. Equality, on the other hand, is a term of type $\Omega^{N\times N}$, where Ω is a certain type, the type of logical propositions. And the terms of *that* type are propositions in the way we have been discussing them so far. For example, $x = x$ is a term of type Ω with a free variable x of type N, and it is itself a proposition, which has a term in it, the proof of the fact that $x = x$. So we have three levels where we previously had only two (types and terms): the types of "big category" like N or Ω or $\Omega^{N\times N}$, which do not express a statement; the terms of type Ω, or propositions, which are themselves types of some kind, and express logical statements in the way we saw up to now, and the terms of those types, which are proofs. The top level category does not have anything remarkable about it: it is a bicartesian closed category with at least two types, N and Ω. It is the type Ω which is magical: its terms (with any, fixed, kind of free variables) themselves form a bicartesian closed category. We then say that Ω is an internal bicartesian closed category (quite a mouthful!), or a bicartesian closed category object. It is this framework that will allow us to define quantifiers in a satisfactory fashion. And note that as an extra bonus we get propositions with free variables in Ω^N and similar types — in other words, we automatically get a higher order logic where quantification is not only on natural numbers but on sets of naturals.

(18) Let $\mathcal{C}$ be a bicartesian closed category. We say that an object Ω of $\mathcal{C}$ is endowed with the structure of an internal bicartesian closed category when for every object U of $\mathcal{C}$ the set $\Omega(U) := \mathrm{Hom}(U, \Omega)$ (of terms of type Ω with a free variable of type U) is endowed with the structure of a bicartesian closed category, and this, in a functorial way (which means that whenever $U \to U'$ is a morphism, the map of sets obtained by composition, $\Omega(U') \to \Omega(U)$, is a morphism of bicartesian closed categories, in other words it is a functor that preserves the operations $\Rightarrow$, $\top$, $\wedge$, $\bot$ and $\vee$). Or, to say things differently, an internal bicartesian closed category is a functor to the category of bicartesian closed categories that is *representable* by an object Ω of $\mathcal{C}$. (Note that we are not imposing that the arrows of the internal category be representable: only the objects.) When P and Q are terms of type Ω with free variables of types $U_1, \ldots, U_n$ (that is, morphisms $U_1 \times \cdots \times U_n \to \Omega$), we can then define $P \Rightarrow Q$ which is another term of type Ω with the same free variables (note that we can always arrange, by adding dummy free variables, to be in that case). Similarily, we have $P \wedge Q$, $P \vee Q$ and $\top$ and $\bot$ (the two last have any free variables you want: we shall carelessly identify all the terms $\top : U_1 \times \cdots \times U_n \to \Omega$). We also have proofs between these terms of type Ω: if $P, Q : U \to \Omega$ are terms of type Ω with U free (we omit the detailed structure), we have the morphisms $x : P \vdash Q$ from P to Q within the category structure of $\Omega(U)$. We then speak of proofs of Q on the assumption that P, but we may also add U as an assumption (or free variable type), even though that is implicit since it is free in P itself. We have two kinds of composition: first, given propositions $P, Q, R : U \to \Omega$ and proofs $P \vdash Q$ and $Q \vdash R$, we may construct a proof $P \vdash R$ by composition within $\Omega(U)$; second, given propositions $P, Q : U' \to \Omega$, a proof $P \vdash Q$ and a morphism $f : U \to U'$ we may use the functoriality of our constructions to yield a proof $Pf \vdash Qf$ (this operation we call variable substitution because it corresponds to changing the free variable of type U' in P and Q to a free variable of type U by means of the morphism f). Abstraction (the λ operation) is also possible, both internally (inside the category $\Omega(U)$) and externally (in the category $\mathcal{C}$). For example, consider the morphism $\Omega \to \Omega$ defined by $P \Rightarrow P$ where P is the free variable of type Ω (as a side note, this morphism is well defined essentially by the Yoneda lemma, which states that a morphism between representable functors is the same thing as a morphism between the objects that represent them). One can abstract it externally to a morphism $1 \to \Omega^\Omega$ defined as $\lambda[P : \Omega].(P \Rightarrow P)$. On the other hand, one can consider the identity morphism (inside $\Omega(\Omega)$!) $P \vdash P$, which by internal abstraction gives $(\lambda[x : P].x){:}\, \top \vdash (P \Rightarrow P)$, with free variable P of type Ω. One might then want to perform abstraction of P in this term, but this is neither internal abstraction (one is abstracting away a variable in $\mathcal{C}$) nor external abstraction (the abstracted term resides in the internal category). It is the *mixed* abstraction of which we shall talk in a second: let us just say that we will get $\lambda[P : \Omega].\lambda[x : P].x$ which is a proof of $\forall(P : \Omega).(P \Rightarrow P)$, the latter being a proposition (in $\Omega(1)$, i.e.with no free variables) of type Ω.

(19) (The Universal Property of $\forall$.) Suppose $Q_x{:}\, U \times V \to \Omega$ is a term of type Ω with free variables x of type U and others of type V. We then define $\forall(x : U).Q_x$ to be the term of type Ω with free variable of type V (i.e.the morphism $V \to \Omega$) that satisifies the universal property that we are about to describe, if it exists. The universal property concerns $\forall(x : U).Q_x$ together with its projection which is a morphism $p_y{:}\, \forall(x : U).Q_x \vdash Q_y$ (in the category $\Omega(U \times V)$, where we have added a fictitious free variable of type U to $\forall(x : U).Q_x$). It states: for any term $P{:}\, V \to \Omega$ and any proof $u_x{:}\, P \vdash Q_x$ (in the category $\Omega(U \times V)$, where we have added a fictitious free variable of type U to P), there exists a unique proof $\lambda[x : U].u_x$ of $\forall(x : U).Q_x$ from the assumption P which composed with the projection p_y of Q_y from the assumption $\forall(x : U).Q_x$ gives precisely the proof u_y we started from. (Note that in all this the index x or y on terms is just to help remember what has a free variable. But Q_x and Q_y are the same thing, just a morphism $U \times V \to \Omega$.) In other words, $\forall(x : U).Q_x$ is universal for those terms of type Ω with free variable of type V (the important thing is that there is no longer the free variable x of type U) "which prove Q_y".

Now consider Q_x and P as above. If $u{:}\, P \vdash \forall(x : U).Q_x$ and $f{:}\, W \to U$ is a term of type U with free variable of type W, then we write uf for the proof $P \vdash Q_f$ (where Q_f stands for the term $W \times V \to \Omega$ obtained by composing $Q{:}\, U \times V \to \Omega$ with f) obtained by composing u with the projection $p_f{:}\, \forall(x : U).Q_x \vdash Q_f$. While this may sound arcane, what we're doing is just applying a proof that $\forall(x : U).Q_x$ to a term f of type U to get a proof of Q_f, which is reasonable. Note that with this notation, the projection $p_y{:}\, \forall(x : U).Q_x \Rightarrow Q_y$ is simply $\lambda[h : \forall(x : U).Q_x].hy$.

(The Universal Property of $\exists$.) The universal property of $\exists$ is similar. Suppose $P_x{:}\, U \times V \to \Omega$ is a term of type Ω with free variables x of type U and others of type V. Then $\exists(x : U).P_x$ is of type Ω and has only a free variables of type V, comes with a proof $q_y{:}\, P_y \vdash \exists(x : U).P_x$ such that for any term Q of type Ω that has only a free variable of type V and any proof u_x of Q from P_x, there exists a unique proof $(x?u_x)$ of Q from $\exists(x : U).P_x$ (x is not free) which composed with the proof q_y of $\exists(x : U).P_x$ from P_y gives precisely the proof u_y we started from.

If P_x is as above, and $f{:}\, W \to U$ is a term of type U with free variable of type W, and $Q{:}\, V \to \Omega$ is a term of type Ω with free variable of type V, and $u_f{:}\, Q \vdash P_f$ is a proof of P_f from Q, then we write (f, u_f) for the composition of the proof q_f of $\exists(x : U).P_x$ from P_f and the proof u_f of P_f from Q. Also, concerning the notation $(x?u_x)$ we introduced above (which says something like "since there exists such an object, call it x"), we make the convention that when writing u_x, the variable of type $\exists(x : U).P_x$ which is being assumed will be written with an extra subscript x, because it then appears of type P_x (if this sounds obscure, compare with a case construction like $\lambda[x : A \vee A].(x_{\leftarrow}|x_{\rightarrow})$ (of type $A \vee A \Rightarrow A$) in which, although x has type $A \vee A$, it is used with type A in the parentheses, which we recall with the arrow subscript).

Note that the objects $\forall(x:U).R_x$ and $\exists(x:U).R_x$ satisfying the universal properties we have just given may or may not exist. When they exist for any type U and any term R_x of type Ω with free variable of type U, we say that Ω is a complete internal bicartesian closed category (whew!). We now make this assumption.

(20) We now give a few examples to illustrate the notations. First of all, let us prove that if U and V are types then

$$\forall(f:\Omega^{U\times V}).(\exists(x:U).\forall(y:V).fxy) \Rightarrow (\forall(y:V).\exists(x:U).fxy)$$

To this effect, we take f a free variable of type $\Omega^{U\times V}$ and h an assumption that $\exists(x:U).\forall(y:V).fxy$. Now if h_x is an assumption that $\forall(y:V).fxy$ (where the free variables are x of type U and f of type $\Omega^{U\times V}$) then $h_x y$ is a proof of fxy (where we have added one more free variable, y, of type V), so that $(x, h_x y)$ is a proof of $\exists(x:U).fxy$ and $\lambda[y:V].(x, h_x y)$ is a proof of $\forall(y:V).\exists(x.U).fxy$. Now this was based on the assumption h_x that $\forall(y:V).fxy$, and therefore $(x?\lambda[y:V].(x, h_x y))$ is a proof of the same based on the assumption h that $\exists(x:U).\forall(y:V).fxy$. Therefore $\lambda h.(x?\lambda[y:V].(x, h_x y))$ is a proof of the implication, and finally the proof we wanted is

$$\lambda[f:\Omega^{U\times V}].\lambda[h:\exists(x:U).\forall(y:V).fxy].(x?\lambda[y:V].(x, h_x y))$$