**Quant FAQs**

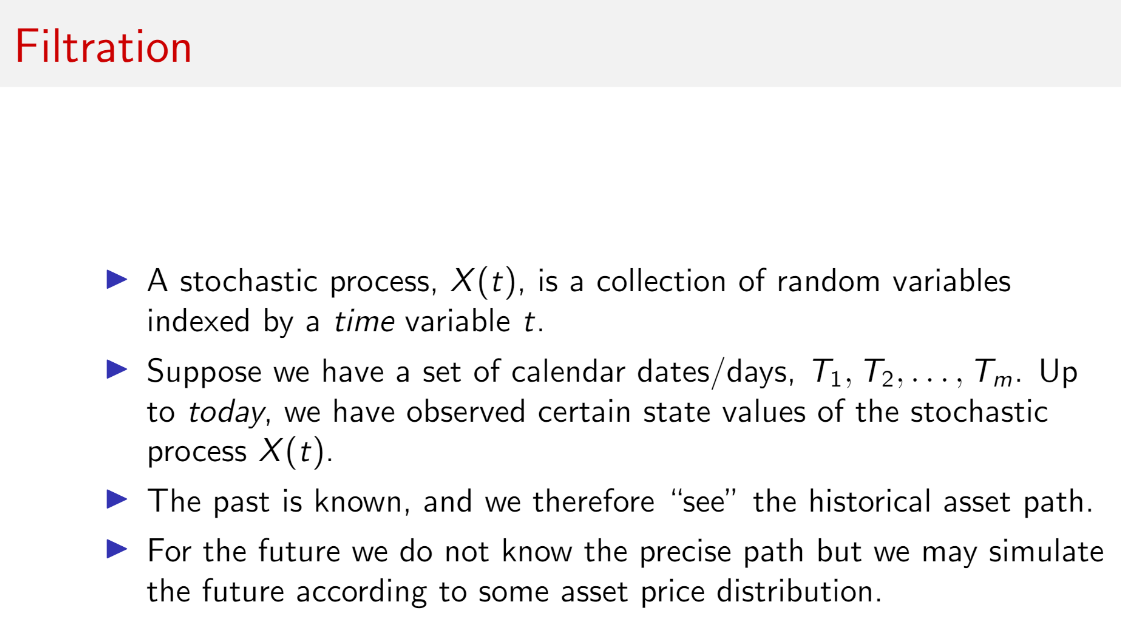
# Stoch Calculus

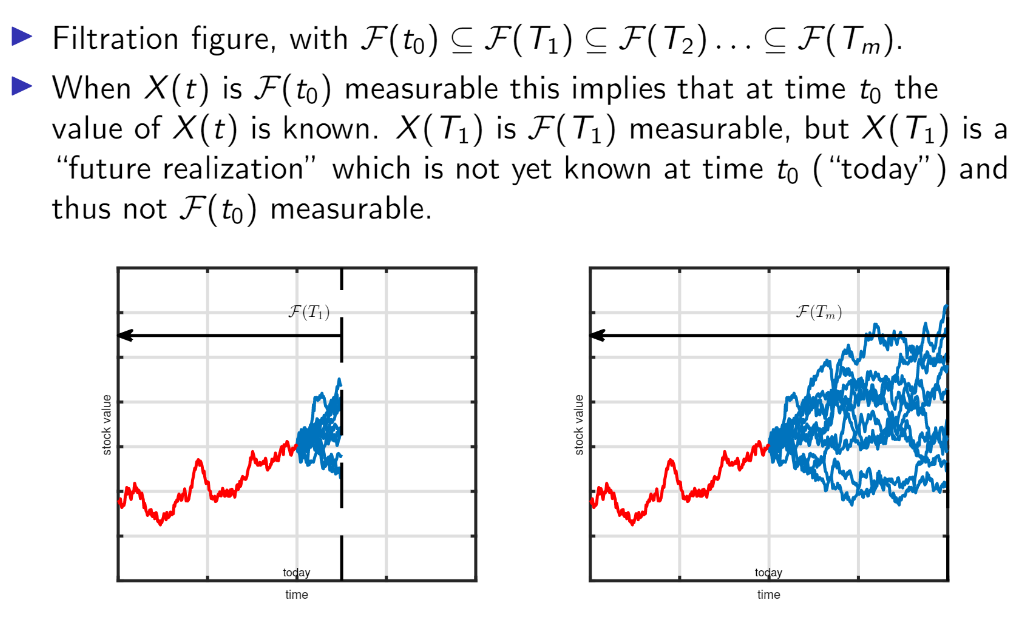
## Filtration

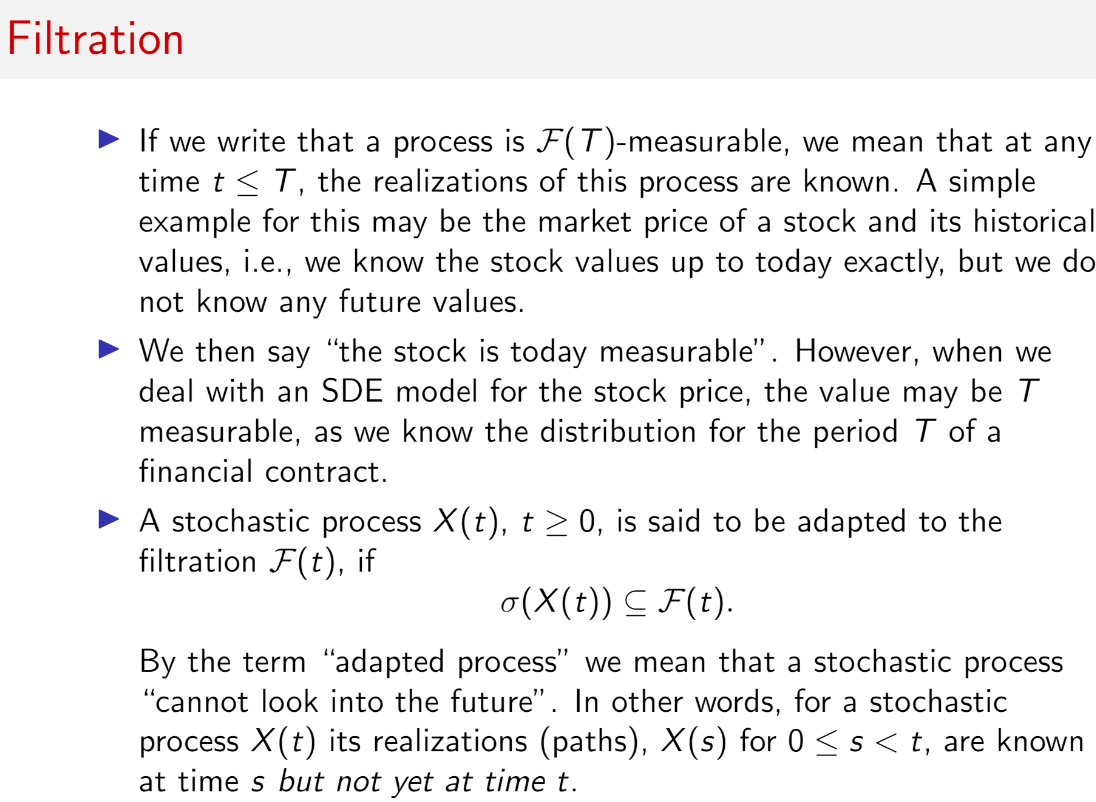
**Question:** What is Filtration? Why is it important in stochastic processes?

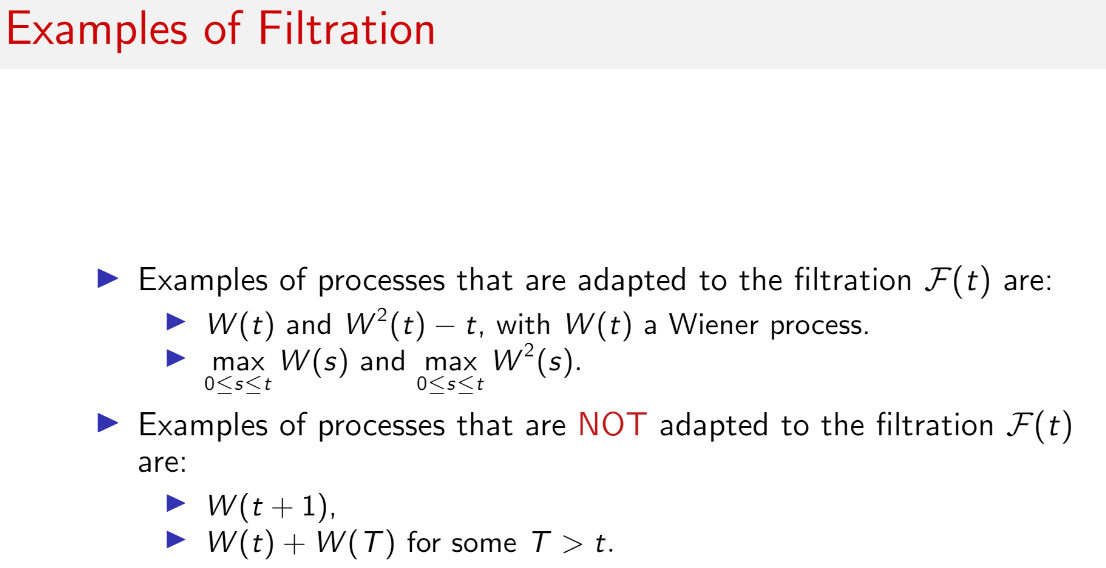
**My Answer:**

“Knowledge that we have at given time t” -> what happened up to this point









## Stochastic process

**Question: What is a stochastic process in layman's terms?**

**My Answer:**

Anything completely random is not important. If there is no pattern in its of no use.

Even though the toss of a fair coin is random but there is a pattern that given sufficiently large number of trails you will get half of the times as heads. This is a useful pattern.

The pollen in water will follow Brownian motion. Even though it appears that motion of individual pollen is independent of its previous direction, overall there is pattern observed with that kind of motion.

In rolling of two dice you may find at any given time the sum of outcome of two dice is random but if you take a series of events ( roll n number of times ) you may observe that 9 and 12 appear less times than 10 and 11. There is a pattern in them even though the individuals are random.

These types of process ( series of events) are called stochastic processes. The individual events are random but there is inherently a pattern within them.

That is important.

## Stationarity

**Question: What is a stationarity?**

**My Answer:**

A stochastic process is stationary if:

* Has a constant mean
* Constant variance
* Constant covariance with its lags

## Markov property and Martingales

**Question: What does Markov property imply? What is a martingale?**

**My Answer:**

* **Markov property:** the process is Markov if its expectation is not dependent on any previous values in a sequence
* **Martingale:** the conditional expectation of a sequence of partial equals a present value

A sensible way to introduce the **Markov property** is through a sequence of random variables *Zi*, which can take one of two values from the set {1,−1}. This is known as a coin toss. We can calculate the expectations of Zi:

*E(Zi)=0,E(Zi2)=1,E(ZiZk)=0*

**The key point is that the expectation of Zi has no dependence on any previous values within the sequence.** Let us take the partial sums of our random variables within our coin toss, which we will denote by Si:

*Si=∑k=1iZi*

We can now calculate the expectations of our partial sums, using the linearity of the expectation operator:

*E(Si)=0,E(Si2)=E(Z12+2Z1Z2+...)=i*

We see that, again, there is no dependence on the expectation of Si of any previous value within the sequence of partial sums. We can extend this to discuss conditional expectation. Conditional expectation is the expectation of a random variable with respect to some conditional probability distribution. Hence, we can ask that if i=4 (i.e. we carry out four coin tosses), what does this mean for the expectation of S5?

*E(S5|Z1,Z2,Z3,Z4)=S4*

That is, the expected value of Si is only dependent upon the previous value Si−1, not on any values prior to that. This is known as the Markov Property. Essentially, there is no memory of past events beyond the point our variable is currently at within the sequence. Nearly all financial models discussed in these articles will possess the Markov property.

An additional property that holds for our sequence of partial sums is the **Martingale property**. It states **that the conditional expectation of the sequence of partial sums, Si is simply the current value**:

*E(Si|Sk,k<i)=Sk*

Essentially, the martingale property ensures that in a "fair game", knowledge of the past will be of no use in predicting future winnings.

These properties will be of fundamental importance in regard to defining Brownian motion, which will later be used as a model for an asset price path.

## Brownian motion / Wiener process

**Question: What is Brownian Motion?**

**My Answer:** It is a stochastic process that can be characterized by**:**

* **Stationary independent** increments
* Increments that are **normally** distributed
* It is a **continuous** process
* It is a **finite process** (scaling of variance in time)
* It is **Markov** (no dependence on the past)
* It is a **martingale** (its conditional expected value equal the current value)

**Short Answer**

Brownian Motion is a stochastic process with stationary independent normally distributed increments and which also has continuous sample paths. It is the most common stochastic building block for random walks in finance.

**Example**

Pollen in water, smoke in a room, pollution in a river, are all examples of Brownian motion. And this is the common model for stock prices as well.

**Long Answer**

Brownian motion (BM) is named after the Scottish botanist who first described the random motions of pollen grains suspended in water. The mathematics of this process were formalized by Bachelier, in an option-pricing context, and by Einstein. The mathematics of BM is also that of heat conduction and diffusion.

Mathematically, BM is a continuous, stationary, stochastic process with independent normally distributed increments.

If *Wt* is the BM at time *t* then for every *t*, *τ* ≥ 0, *Wt*+*τ* −*Wt* is independent of {*Wu* : 0 ≤ *u* ≤ *t*}, and has a normal distribution with zero mean and variance *τ* .

The important properties of BM are as follows.

• ***Finiteness***: the scaling of the variance with the time step is crucial to BM remaining finite.

• ***Continuity***: the paths are continuous, there are no discontinuities. However, the path is fractal, and not differentiable anywhere.

***Markov:*** the conditional distribution of *Wt* given information up until *τ < t* depends only on *Wτ* .

• ***Martingale***: given information up until *τ < t* the conditional expectation of *Wt* is *Wτ* .

• ***Quadratic*** *variation*: if we divide up the time 0 to *t* in a partition with *n* + 1 partition points *ti* = *it/n* then



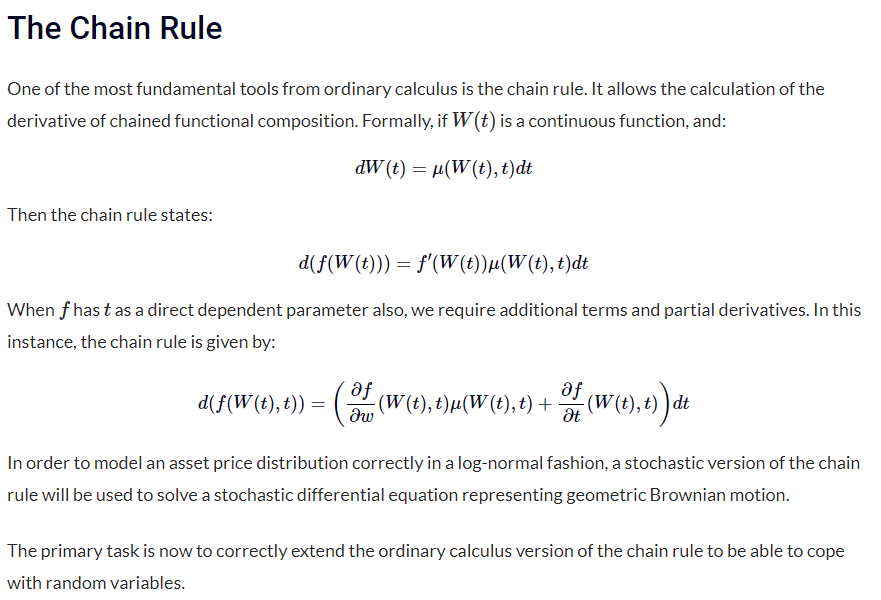
• ***Normality***: Over finite time increments *ti*−1 to *ti*,*Wti*−*Wti*−1 is normally distributed with mean zero and variance *ti* − *ti*−1.

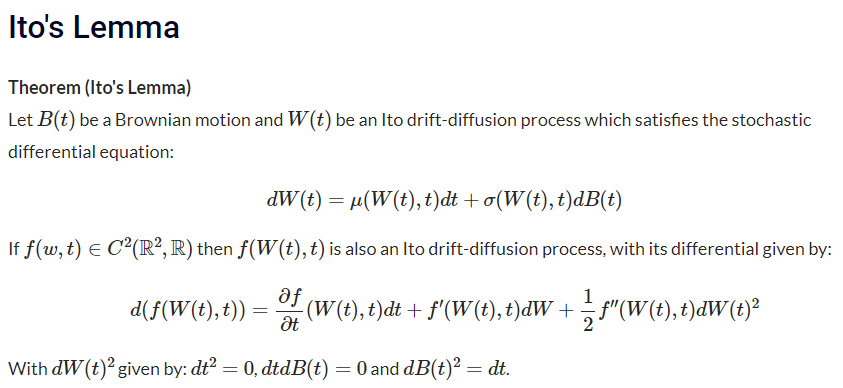
BM is a very simple yet very rich process, extremely useful for representing many random processes especially those in finance. Its simplicity allows calculations and analysis that would not be possible with other processes. For example, in option pricing it results in simple closed-form formula for the prices of vanilla options. It can be used as a building block for random walks with characteristics beyond those of BM itself. For example, it is used in the modelling of interest rates via mean-reverting random walks. Higher-dimensional versions of BM can be used to represent multi-factor random walks, such as stock prices under stochastic volatility.

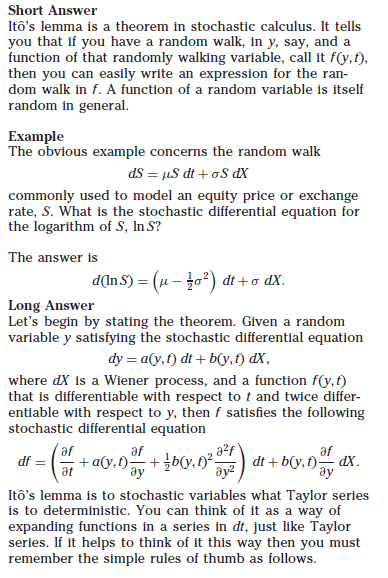
One of the unfortunate features of BM is that it gives returns distributions with tails that are unrealistically shallow. In practice, asset returns have tails that are much fatter than that given by the normal distribution of BM. There is even some evidence that the distribution of returns have infinite second moment. Despite this, and the existence of financial theories that do incorporate such fat tails, BM motion is easily the most common model used to represent random walks in finance.

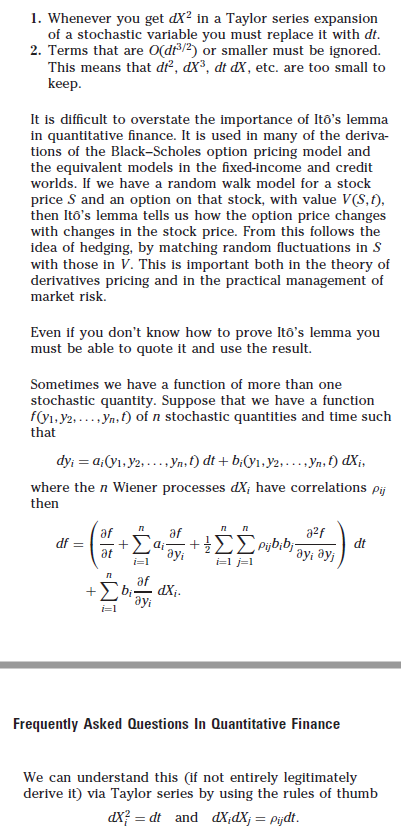
## Geometric Brownian Motion

## Ito Lemma









## First Fundamental Asset Pricing Theorem

## What is arbitrage?

|  |  |  |  |
| --- | --- | --- | --- |
| Arbitrage is making a sure profit in excess of the risk free rate of return. In the language of quantitative finance we can say an arbitrage opportunity is a portfolio of zero value today which is of positive value in the future with positive probability and of negative value in the future with zero probability. The assumption that there are no arbitrage opportunities in the market is fundamental to classical finance theory. This idea is popularly known as ‘there’s no such thing as a free lunch.’ | | | |
|  | |  |
|  | Example | | |
|  | • An at-the-money European call option with a strike of$100 and an expiration of six months is worth $8. | | |
|  | • A European put with the same strike and expiration is worth $6. | | |
|  | • There are no dividends on the stock and a six-month zero-coupon bond with a principal of $100 is worth $97. | | |
|  | • Buy the call and a bond, sell the put and the stock, which will bring in $ − 8 − 97 + 6 + 100 = $1. | | |
|  | • At expiration this portfolio will be worthless regardless of the final price of the stock. | | |
|  | • You will make a profit of $1with no risk. This is arbitrage. It is an example of the violation of put-call parity. | | |
|  | Long Answer | | |
|  | The principle of no arbitrage is one of the foundations of classical finance theory. In derivatives theory it is assumed during the derivation of the binomial model option pricing algorithm and in the Black–Scholes model. In these cases it is rather more complicated than the simple example given above. In the above example set up a portfolio that gave us an immediate profit, and that portfolio did not have to be touched until expiration. This is a case of a static arbitrage. Another special feature of the above example is that it does not rely on any assumptions about how the stock price behaves. So the example is that of model-independent arbitrage. However, when deriving the famous option pricing models we rely on a dynamic strategy, called delta hedging, in which a portfolio consisting of an option and stock is constantly adjusted by purchase or sale of stock in a very specific manner. Now we can see that there are several types of arbitrage that we can think of. Here is a list and description of the most important. | | |
|  | • A static arbitrage is an arbitrage that does not require rebalancing of positions | | |
|  | • A dynamic arbitrage is an arbitrage that requires trading instruments in the future, generally contingent on market states | | |
|  | • A statistical arbitrage is not an arbitrage but simply a likely profit in excess of the risk-free return (perhaps even suitably adjusted for risk taken) as predicted by past statistics | | |
|  | • Model-independent arbitrage is an arbitrage which does not depend on any mathematical model of financial instruments to work. For example, an exploitable violation of put-call parity or a violation of the relationship between spot and forward prices, or between bonds and swaps | | |
|  | • Model-dependent arbitrage does require a model. For example, options mispriced because of incorrect volatility estimate. To profit from the arbitrage you need to delta hedge and that requires a model. | | |
|  | Not all apparent arbitrage opportunities can be exploited in practice. If you see such an opportunity in quoted prices on a screen in front of you then you are likely to find that when you try to take advantage of them they just evaporate. Here are several reasons for this. | | |
|  | • Quoted prices are wrong or not tradeable | | |
|  | • Option and stock prices were not quoted synchronously | | |
|  | • There is a bid-offer spread you have not accounted for | | |
|  | • Your model is wrong, or there is a risk factor you have not accounted for | | |

## Delta hedging

|  |
| --- |
|  |
| **Delta hedging**  One of the building blocks of derivatives theory is delta hedging. This is the theoretically perfect elimination of all risk by using a very clever hedge between the option and its underlying.  Delta hedging exploits the perfect correlation between the changes in the option value and the changes in the stock price. This is an example of ‘dynamic’ hedging; the hedge must be continually monitored and frequently adjusted by the sale or purchase of the underlying asset. Because of the frequent rehedging, any dynamic hedging strategy is going to result in losses due to transaction costs. In some markets this can be very important. The ‘underlying’ in a delta-hedged portfolio could be a traded asset, a stock for example, or it could be another random quantity that determines a price such as a risk of default. If you have two instruments depending on the same risk of default, you can calculate the sensitivities, the deltas, of their prices to this quantity and then buy the two instruments in amounts inversely proportional to these deltas (one long, one short). This is also delta hedging. If two underlyings are very highly correlated you can use one as a proxy for the other for hedging purposes. You would then only be exposed to basis risk. Be careful with this because there may be times when the close relationship breaks down. If you have many financial instruments that are uncorrelated with each other then you can construct a portfolio with much less risk than any one of the instruments individually. With a large such portfolio you can theoretically reduce risk to negligible levels. Although this isn’t strictly hedging it achieves the same goal. | |
|  |

## Risk neutral measure

|  |  |
| --- | --- |
| Risk-neutral valuation means that you can value options in terms of their expected payoffs, discounted from expiration to the present, assuming that they grow on average at the risk-free rate . | |
|  | *Option value = Expected present value of payoff(under a risk-neutral random walk).* |
|  | Therefore the real rate at which the underlying grows on average doesn’t affect the value. Of course, the volatility, related to the standard deviation of the underlying’s return, does matter. In practice, it’s usually much, much harder to estimate this average growth than the volatility, so we are rather spoiled in derivatives, that we only need to estimate the relatively stable parameter, volatility. The reason that this is true is that by hedging an option with the underlying we remove any exposure to the direction of the stock, whether it goes up or down ceases to matter. By eliminating risk in this way we also remove any dependence on the value of risk. End result is that we may as well imagine we are in a world in which no one values risk at all, and all tradeable assets grow at the risk-free rate on average. For any derivative product, as long as we can hedge it dynamically and perfectly (supposing we can as in the case of known, deterministic volatility and no defaults) the hedged portfolio loses its randomness and behaves like a bond. | |
|  | 2I should emphasize the word ‘relatively.’ Volatility does varying reality, but probably not as much as the growth rate. | |
|  | **Example** | |
|  | A stock whose value is currently $44.75 is growing on average by 15% per annum. Its volatility is 22%. The interest rate is 4%. You want to value a call option with a strike of $45, expiring in two months’ time. What can you do? First of all, the 15% average growth is totally irrelevant. The stock’s growth and therefore its real direction does not affect the value of derivatives. What you can do is simulate many, many future paths of a stock with an average growth of 4% per annum, since that is the risk free interest rate, and a 22% volatility, to find out where it may be in two months’ time. Then calculate the call payoff for each of these paths. Present value each of these back to today, and calculate the average over all paths. That’s your option value. | |
|  | **Long Answer** | |
|  | Risk-neutral valuation of derivatives exploits the perfect correlation between the changes in the value of an option and its underlying asset. As long as the underlying is the only random factor then this correlation should be perfect. So if an option goes up in value with a rise in the stock then a long option and sufficiently short stock position shouldn’t have any random fluctuations, therefore the stock hedges the option. The resulting portfolio is risk free. Of course, you need to know the correct number of the stock to sell short. That’s called the ‘delta’ and usually comes from a model. Because we usually need a mathematical model to calculate the delta, and because quantitative finance models are necessarily less than perfect, the theoretical elimination of risk by delta hedging is also less than perfect in practice. There are several such imperfections with risk-neutral valuation. First, it requires continuous rebalancing of the hedge. Delta is constantly changing so you must always be buying or selling stock to maintain a risk-free position. Obviously, this is not possible in practice. Second, it hinges on the accuracy of the model. The underlying has to be consistent with certain assumptions, such as being Brownian motion without any jumps, and with known volatility. One of the most important side effects of risk-neutral pricing is that we can value derivatives by doing simulations of the risk-neutral path of underlyings, to calculate payoffs for the derivatives. These payoffs are then discounted to the present, and finally averaged. This average that we find is the contract’s fair value. Here are some further explanations of risk-neutral pricing. | |
|  | **Explanation 1:** If you hedge correctly in a Black–Scholes world then all risk is eliminated. If there is no risk then we should not expect any compensation for risk. We can therefore work under a measure in which everything grows at the risk-free interest rate. | |
|  | **Explanation 2**: If the model for the asset is dS = μS dt +σS dX then the μs cancel in the derivation of the Black–Scholes equation. | |
|  | **Explanation 3:** Two measures are equivalent if they have the same sets of zero probability. Because zero Probability sets don’t change, a portfolio is an arbitrage under one measure if and only if it is one under all equivalent measures. Therefore a price is non-arbitrageable in the real world if and only if it is non-arbitrageable in the risk-neutral world. The risk-neutral price is always nonarbitrageable. If everything has a discounted asset price process which is a martingale then there can be no arbitrage. So if we change to a measure in which all the fundamental assets, for example the stock and bond, are martingales after discounting, and then define the option price to be the discounted expectation making it into a martingale too, we have that everything is a martingale in the risk-neutral world. Therefore there is no arbitrage in the real world. | |
|  | **Explanation 4:** If we have calls with a continuous distribution of strikes from zero to infinity then we can synthesize arbitrarily well any payoff with the same expiration. But these calls define the risk-neutral probability density function for that expiration, and so we can interpret the synthesized option in terms of risk-neutral random walks. When such a static replication is possible then it is model independent, we can price complex derivatives in terms of vanillas. (Of course, the continuous distribution requirement does spoil this argument to some extent.) It should be noted that risk-neutral pricing only works under assumptions of continuous hedging, zero transaction costs, continuous asset paths, etc. Once we move away from this simplifying world we may find that it doesn’t work. | |

## Girsanov theorem

|  |  |
| --- | --- |
|  | |
|  | **Short Answer**  Girsanov’s theorem is the formal concept underlying the change of measure from the real world to the risk neutral world. We can change from a Brownian motion with one drift to a Brownian motion with another. | | | |
|  | **Example** | | |
|  | The classical example is to start with dS = μS dt + σS dWt with W being Brownian motion under one measure (the real-world measure) and converting it to dS = rS dt + σS d ˜Wt under a different, the risk-neutral, measure. | |
|  | **Long Answer** | | |
|  | First a statement of the theorem. Let Wt be a Brownian motion with measure P and sample space \_. Ifγt is a previsible process satisfying the constrain tEP\_exp\_12\_ T0 γ 2t\_\_< ∞ then there exists an equivalent measure Q on \_ such that˜Wt = Wt +\_ t0γsds is a Brownian motion. It will be helpful if we explain some of the more technical terms in this theorem. | | |
|  | **Sample space**: All possible future states or outcomes. | | |
|  | **(Probability) Measure**: In layman’s terms, the measure gives the probabilities of each of the outcomes in the sample space. | | |
|  | **Previsible**: A previsible process is one that only depends on the previous history. | | |
|  | **Equivalent**: Two measures are equivalent if they have the same sample space and the same set of ‘possibilities.’ Note the use of the word possibilities instead of probabilities. The two measures can have different probabilities for each outcome but must agree on what is possible. Another way of writing the above is in differential form d ˜Wt = dWt + γt dt. One important point about Girsanov’s theorem is its converse, that every equivalent measure is given by a drift change. This implies that in the Black–Scholes world there is only the one equivalent risk-neutral measure. If this were not the case then there would be multiple arbitrage-free prices. For many problems in finance Girsanov theorem is not necessarily useful. This is often the case in the world of equity derivatives. Straightforward Black–Scholes does not require any understanding of Girsanov. Once you go beyond basic Black–Scholes it becomes more useful. For example, suppose you want to derive the valuation partial differential equations for options under stochastic volatility. The stock price follows the real world processes, P, | | |
|  | dS = μS dt + σS dX1 | | |
|  | and | | |
|  | dσ = a(S, σ, t)dt + b(S, σ, t)dWX2, | | |
|  | where dX1 and dX2 are correlated Brownian motions with correlation ρ(S, σ, t). | | |
|  | Using Girsanov you can get the governing equation in three steps: | | |
|  | 1. Under a pricing measure Q, Girsanov plus the fact that S is traded implies that dX1 = d˜X1 − μ – rσdt and dX2 = d˜X2 − λ(S, σ, t) dt, where λ is the market price of volatility risk | | |
|  | 2. Apply Itˆo’s formula to the discounted option price V(S, σ, t) = e−r(T−t)F(S, σ, t), expanding under Q, using the formulæ for dS and dV obtained from the Girsanov transformation | | |
|  | 3. Since the option is traded, the coefficient of the dt term in its Itˆo expansion must also be zero; this yields the relevant equation Girsanov and the idea of change of measure are particularly important in the fixed-income world where practitioners often have to deal with many different measures at the same time, corresponding to different maturities. This is the reason for the popularity of the BGM model and its ilk. | | |

## Black Scholes

**Short Answer**

BS model is very robust. You can drop quite a few of the assumptions underpinning Black–Scholes and it won’t fall over.

**Example**

Transaction costs? Simply adjust volatility.

Time dependent volatility? Use root-mean-square-average volatility instead.

Interest rate derivatives? Black ’76 explains how to use the Black–Scholes formula in situations where it wasn’t originally intended.

**Long Answer**

Here are some assumptions that seems crucial to the whole Black–Scholes model, and what happens when you drop those assumptions.

**Hedging is continuous:** If you hedge discretely it turns out that Black–Scholes is right on average. In other words sometimes you lose because of discrete hedging, sometimes you win, but on average you break even. And Black–Scholes still applies.

**There are no transaction costs:** If there is a cost associated with buying and selling the underlying for hedging this can be modelled by a new term in the Black–Scholes equation that depends on gamma. And that term is usually quite small. If you rehedge at fixed time intervals then the correction is proportional to the absolute value of the gamma, and can be interpreted as simply a correction to volatility in the standard Black–Scholes formula. So instead of pricing with a volatility of 20%, say, you might use 17% and 23% to represent the bid offer spread dues to transaction costs.

**Volatility is constant:** If volatility is time dependent then the Black–Scholes formula are still valid as long as you plug in the ‘average’ volatility over the remaining life of the option. Here average means the root-mean-square average since volatilities can’t be added but variances can. Even if volatility is stochastic we can still use basic Black–Scholes formula provided the volatility process is independent of, and uncorrelated with, the stock price. Just plug the average variance over the option’s lifetime, conditional upon its current value, into the formula.

**There are no arbitrage opportunities:** Even if there are arbitrage opportunities because implied volatility is different from actual volatility you can still use the Black–Scholes formula to tell you how much profit you can expect to make, and use the delta formula to tell you how to hedge. Moreover, if there is an arbitrage opportunity and you don’t hedge properly, it probably won’t have that much impact on the profit you expect to make.

**The underlying is lognormally distributed:** The Black–Scholes model is often used for interest-rate products which are clearly not lognormal. But this approximation is often quite good, and has the advantage of being easy to understand. This is the model commonly referred to as Black ’76.

**There are no costs associated with borrowing stock for going short:** Easily accommodated within a Black–Scholes model, all you need to do is make an adjustment to the risk-neutral drift rate, rather like when you have a dividend.

**Returns are normally distributed: Thanks** to near-continuous hedging and the Central Limit Theorem all you really need is for the returns distribution to have a finite variance, the precise shape of that distribution, its skew and kurtosis, don’t much matter. Black–Scholes is a remarkably robust model.

## What is put-call parity?

**Short Answer**

Put-call parity is a relationship between the prices of a European-style call option and a European-style put option, as long as they have the same strike and expiration:

Call price − Put price = Stock price− Strike price (present valued from expiration).

**Example**:

Stock price is $98, a European call option struck at $100 with an expiration of nine months has a value of $9.07. The nine-month, continuously compounded, interest rate is 4.5%. What is the value of the put option with the same strike and expiration? By rearranging the above expression we find

Put price = 9.07 − 98 + 100 e−0.045×0.75 = 7.75.

The put must therefore be worth $7.75.

**Long Answer**

This relationship, C − P = S −K e−r(T−t), between European calls (value C) and puts (value P) with the same strike (K) and expiration (T) valued at time t is a result of a simple arbitrage argument. If you buy a call option, at the same time write a put, and sell stock short, what will your payoff be at expiration? If the stock is above the strike at expiration you will have S − K from the call, 0 from the put and −S from the stock. A total of −K. If the stock is below the strike at expiration you will have 0 from the call, −S again from the stock, and −(K − S) from the short put. Again a total of −K. So, whatever the stock price is at expiration this portfolio will always be worth −K, a guaranteed amount. Since this amount is guaranteed we can discount it back to the present. We must have C − P − S = −K e−r(T−t).This is put-call parity. Another way of interpreting put-call parity is in terms of implied volatility. Calls and puts with the same strike and expiration must have the same implied volatility. The beauty of put-call parity is that it is a model independent relationship. To value a call on its own we need a model for the stock price, in particular its volatility. The same is true for valuing a put. But to value a portfolio consisting of a long call and a short put (or vice versa), no model is needed. Such model independent relationships are few and far between in finance. The relationship between forward and spot prices is one, and the relationships between bonds and swaps is another. In practice options don’t have a single price, they have two prices, a bid and an offer (or ask). This means that when looking for violations of put-call parity you must use bid (offer) if you are going short (long) the options. This makes the calculations a little bit messier. If you think in terms of implied volatility then it’s much easier to spot violations of put-call parity. You must look for non-overlapping implied volatility ranges. For example, suppose that the bid/offer on a call is 22%/25% in implied volatility terms and that on a put (same strike and expiration) is 21%/23%. There is an overlap between these two ranges (22–23%) and so no arbitrage opportunity. However, if the put prices were 19%/21% then there would be a violation of put-call parity and hence an easy arbitrage opportunity. Don’t expect to find many (or, indeed, any) of such simple free-money opportunities in practice though. If you do find such an arbitrage then it usually disappears by the time you put the trade on. See Kamara & Miller (1995) for details of the statistics of no-arbitrage violations. When there are dividends on the underlying stock during the life of the options then we must adjust the equation to allow for this. We now find that C − P = S − Present value of all dividends − E e−r(T−t).This, of course, assumes that we know what the dividends will be. If interest rates are not constant then just discount the strike back to the present using the value of a zero coupon bond with maturity the same as expiration of the option. Dividends should similarly be discounted. When the options are American, put-call parity does not hold. This is because the short position could be exercised against you, leaving you with some exposure to the stock price. Therefore you don’t know what you will be worth at expiration. In the absence of dividends it is theoretically never optimal to exercise an American call before expiration, whereas an American put should be exercised if the stock falls low enough.

## Greeks

**Short Answer**

The ‘greeks’ are the sensitivities of derivatives prices to underlyings, variables and parameters. They can be calculated by differentiating option values with respect to variables and/or parameters, either analytically, if you have a closed-form formula, or numerically.

**Example**

Delta, \_ = ∂V∂S , is sensitivity of option price to the stock price. Gamma, \_ = ∂2V∂S2 , is the second derivative of the option price to the underlying stock, it is the sensitivity of the delta to the stock price. These two examples are called greek because they are members of the Greek alphabet. Some sensitivities, such as vega = ∂V∂σ , are still called ‘greek’ even though they aren’t in the Greek alphabet.

**Long Answer**

**Delta** The delta, \_, of an option or a portfolio of option sis the sensitivity of the option or portfolio to the underlying. It is the rate of change of value with respect to the asset:\_ = ∂V∂S. Speculators take a view on the direction of some quantity such as the asset price and implement a strategy to take advantage of their view. If they own options then their exposure to the underlying is, to a first approximation, the same as if they own delta of the underlying. Those who are not speculating on direction of the underlying will hedge by buying or selling the underlying, or another option, so that the portfolio delta is zero. By doing this they eliminate market risk. Typically the delta changes as stock price and time change, so to maintain a delta-neutral position the number of assets held requires continual readjustment by purchase or sale of the stock. This is called rehedging or rebalancing the portfolio, and is an example of dynamic hedging. Sometimes going short the stock for hedging purposes requires the borrowing of the stock in the first place.(You then sell what you have borrowed, buying it back later.) This can be costly, you may have to pay a repo rate, the equivalent of an interest rate, on the amount borrowed.

**Gamma** The gamma, \_, of an option or a portfolio of options is the second derivative of the position with respect to the underlying:\_ = ∂2V∂S2 .Since gamma is the sensitivity of the delta to the underlying it is a measure of by how much or how often a position must be rehedged in order to maintain a delta-neutral position. If there are costs associated with buying or selling stock, the bid-offer spread, for example, then the larger the gamma the larger the cost or friction caused by dynamic hedging. Because costs can be large and because one wants to reduce exposure to model error it is natural to try to minimize the need to rebalance the portfolio too frequently. Since gamma is a measure of sensitivity of the hedge ratio \_ to the movement in the underlying, the hedging requirement can be decreased by a gamma neutral strategy. This means buying or selling more options, not just the underlying.

**Theta** The theta, \_, is the rate of change of the option price with time.\_ = ∂V∂t. The theta is related to the option value, the delta and the gamma by the Black--Scholes equation.

**Speed** The speed of an option is the rate of change of the gamma with respect to the stock price. Speed = ∂3V∂S3 .Traders use the gamma to estimate how much they will have to rehedge by if the stock moves. The stock moves by $1 so the delta changes by whatever the gamma is. But that’s only an approximation. The delta may change by more or less than this, especially if the stock moves by a larger amount, or the option is close to the strike and expiration. Hence the use of speed in a higher-order Taylor series expansion.

**Vega** The vega, sometimes known as zeta or kappa, is a very important but confusing quantity. It is the sensitivity of the option price to volatility. Vega = ∂V∂σ. This is completely different from the other greeks since it is a derivative with respect to a parameter and not a variable. This can be important. It is perfectly acceptable to consider sensitivity to a variable, which does vary, after all. However, it can be dangerous to measure sensitivity to something, such as volatility, which is a parameter and may, for example, have been assumed to be constant. That would be internally inconsistent. As with gamma hedging, one can vega hedge to reduce sensitivity to the volatility. This is a major step towards eliminating some model risk, since it reduces dependence on a quantity that is not known very accurately. There is a downside to the measurement of vega. It is only really meaningful for options having single-signed gamma everywhere. For example it makes sense to measure vega for calls and puts but not binary calls and binary puts. The reason for this is that call and put values (and options with single-signed gamma) have values that are monotonic in the volatility: increase the volatility in a call and its value increases everywhere. Contracts with a gamma that changes sign may have a vega measured at zero because as we increase the volatility the price may rise somewhere and fall somewhere else. Such a contract is very exposed to volatility risk but that risk is not measured by the vega.

**Rho** ρ, is the sensitivity of the option value to the interest rate used in the Black--Scholes formula:ρ = ∂V∂r. In practice one often uses a whole term structure of interest rates, meaning a time-dependent rate r(t). Rho would then be the sensitivity to the level of the rates assuming a parallel shift in rates at all times. Rho can also be sensitivity to dividend yield, or foreign interest rate in a foreign exchange option.

**Charm** The charm is the sensitivity of delta to time.∂2V∂S ∂t.This is useful for seeing how your hedge position will change with time, for example up until the next time you expect to hedge. This can be important near expiration.

**Colour** The colour is the rate of change of gamma withtime.∂3V∂S2 ∂t.

**Vanna** The Vanna is the sensitivity of delta to volatility.∂2V∂S ∂σ. This is used when testing sensitivity of hedge ratio to volatility. It can be misleading at places where gamma is small.

**Vomma or Volga** The Vomma or Volga is the second derivative of the option value with respect to volatility.∂2V∂σ2 .Because of Jensen’s Inequality, if volatility is stochastic the Vomma/Volga measures convexity due to random volatility and so gives you an idea of how much to add(or subtract) from an option’s value.

**Shadow greeks** The above greeks are defined in terms of partial derivatives with respect to underlying, time, volatility, etc. while holding the other variables/parameters fixed. That is the definition of a partial derivative. But, of course, the variables/parameters might, in practice, move together. For example, a fall in the stock derivative has its mathematical meaning of that which is differentiated not its financial meaning as an option .price might be accompanied by an increase in volatility. So one can measure sensitivity as both the underlying and volatility move together. This is called a shadow greek and is just like the concept of a total derivative in, for example, fluid mechanics where one might follow the path of a fluid particle.

## Volatility smile

**Short Answer**

Volatility is annualized standard deviation of returns. Or is it? Because that is a statistical measure, necessarily backward looking, and because volatility seems to vary, and we want to know what it will be in the future, and because people have different views on what volatility will be in the future, things are not that simple.

**Example**

Actual volatility is the σ that goes into the Black–Scholes partial differential equation. Implied volatility is the number in the Black–Scholes formula that makes a theoretical price match a market price.

**Long Answer**

Actual volatility is a measure of the amount of randomness in a financial quantity at any point in time. It’s what Desmond Fitzgerald calls the ‘bouncy, bouncy.’ It’s difficult to measure, and even harder to forecast but it’s one of the main inputs into option pricing models. It’s difficult to measure since it is defined mathematically via standard deviations which requires historical data to calculate. Yet actual volatility is not a historical quantity but an instantaneous one. Realized/historical volatilities are associated with a period of time, actually two periods of time. We might say that the daily volatility over the last sixty days has been 27%. This means that we take the last sixty days’ worth of daily asset prices and calculate the volatility. Let me stress that this has two associated timescales, whereas actual volatility has none. This tends to be the default estimate of future volatility in the absence of any more sophisticated model. For example, we might assume that the volatility of the next sixty days is the same as over the previous sixty days. This will give us an idea of what a sixty-day option might be worth. Implied volatility is the number you have to put into the Black–Scholes option-pricing equation to get the theoretical price to match the market price. Often said to be the market’s estimate of volatility. Let’s recap. We have actual volatility which is the instantaneous amount of noise in a stock price return. It is sometimes modelled as a simple constant, sometimes as time dependent, sometimes as stock and time dependent, sometimes as stochastic and sometimes as a jump process, and sometimes as uncertain, that is, lying within a range. It is impossible to measure exactly, the best you can do is to get a statistical estimate based on past data. But this is the parameter we would dearly love to know because of its importance in pricing derivatives. Some hedge funds believe that their edge is in forecasting this parameter better than other people, and so profit from options that are mispriced in the market. Since you can’t see actual volatility people often rely on measuring **historical or realized volatility**. This is a backward looking statistical measure of what volatility has been. And then one assumes that there is some information in this data that will tell us what volatility will be in the future. There are several models for measuring and forecasting volatility and we will come back to them shortly.

**Implied volatility** is the number you have to put into the Black–Scholes option-pricing formula to get the theoretical price to match the market price. This is often said to be the market’s estimate of volatility. More correctly, option prices are governed by supply and demand. Is that the same as the market taking a view on future volatility? Not necessarily because most people buying options are taking a directional view on the market and so supply and demand reflects direction rather than volatility. But because people who hedge options are not exposed to direction only volatility it looks to them as if people are taking a view on volatility when they are more probably taking a view on direction, or simply buying out-of-the-money puts as insurance against a crash. For example, the market falls, people panic, they buy puts, the price of puts and hence implied volatility goes up. Where the price stops depends on supply and demand, not on anyone’s estimate of future volatility, within reason. Implied volatility levels the playing field so you can compare and contrast option prices across strikes and expirations. There is also forward volatility. The adjective ‘forward’ is added to anything financial to mean values in the future. So forward volatility would usually mean volatility, either actual or implied, over some time period in the future. Finally hedging volatility means the parameter that you plug into a delta calculation to tell you how many of the underlying to sell short for hedging purposes. Since volatility is so difficult to pin down it is a natural quantity for some interesting modelling. Here are some of the approaches used to model or forecast volatility.

**Econometric models:** These models use various forms of time series analysis to estimate current and future expected actual volatility. They are typically based on some regression of volatility against past returns and they may involve autoregressive or moving-average components. In this category are the GARCH type of models. Sometimes one models the square of volatility, the variance, sometimes one uses high-low-open-closed at and not just closing prices, and sometimes one models the logarithm of volatility. The latter seems to be quite promising because there is evidence that actual volatility is lognormally distributed. Other work in this area decomposes the volatility of a stock into components, market volatility, industry volatility and firm-specific volatility. This is similar to CAPM for returns.

**Deterministic models:** The simple Black–Scholes formula assume that volatility is constant or time dependent. But market data suggests that implied volatility varies with strike price. Such market behaviour cannot be consistent with a volatility that is a deterministic function of time. One way in which the Black–Scholes world can be modified to accommodate strike-dependent implied volatility is to assume that actual volatility is a function of both time and the price of the underlying. This is the deterministic volatility (surface) model. This is the simplest extension to the Black–Scholes world that can be made to be consistent with market prices. All it requires is that we have σ(S, t), and the Black–Scholes partial differential equation is still valid. The interpretation of an option’s value as the present value of the expected payoff under a risk-neutral random walk also carries over. Unfortunately the Black–Scholes closed form formula are no longer correct. This is a simple and popular model, but it does not capture the dynamics of implied volatility very well.

**Stochastic volatility:** Since volatility is difficult to measure, and seems to be forever changing, it is natural to mode lit as stochastic. The most popular model of this type is due to Heston. Such models often have several parameters which can either be chosen to fit historical data or, more commonly, chosen so that theoretical prices calibrate to the market. Stochastic volatility models are better at capturing the dynamics of traded option prices better than deterministic models. However, different markets behave differently. Part of this is because of the way traders look at option prices. Equity traders look at implied volatility versus strike, FX traders look at implied volatility versus delta. It is therefore natural for implied volatility curves to behave differently in these two markets. Because of this there have grownup the sticky strike, sticky delta, etc., models, which model how the implied volatility curve changes as the underlying moves.

**Poisson processes:** There are times of low volatility and times of high volatility. This can be modelled by volatility that jumps according to a Poisson process.

**Uncertain volatility**: An elegant solution to the problem of modelling the unseen volatility is to treat it as uncertain, meaning that it is allowed to lie in a specified range but whereabouts in that range it actually is, or indeed the probability of being at any value, are left unspecified. With this type of model we no longer get a single option price, but a range of prices, representing worst-case scenario and best-case scenario.

## Monte Carlo simulation

**Short Answer**

Monte Carlo simulations are a way of solving probabilistic problems by numerically ‘imagining’ many possible scenarios or games so as to calculate statistical properties such as expectations, variances or probabilities of certain outcomes. In finance we use such simulations to represent the future behaviour of equities, exchange rates, interest rates, etc. so as to either study the possible future performance of a portfolio or to price derivatives.

**Example:**

We hold a complex portfolio of investments, we would like to know the probability of losing money over the next year since our bonus depends on us making a profit. We can estimate this probability by simulating how the individual components in our portfolio might evolve over the next year. This requires us to have a model for the random behaviour of each of the assets, including the relationship or correlation between them, if any. Some problems which are completely deterministic can also be solved numerically by running simulations, most famously finding a value for π.

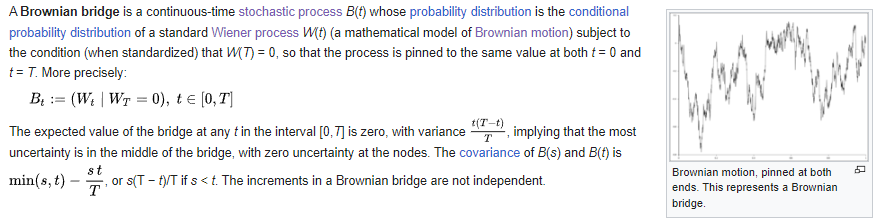
**Long Answer**

It is clear enough that probabilistic problems can be solved by simulations. What is the probability of tossing heads with a coin, just toss the coin often enough and you will find the answer. More on this and its relevance to finance shortly. But many deterministic problem scan also be solved this way, provided you can find aprobabilistic equivalent of the deterministic problem. A famous example of this is Buffon’s needle, a problem and solution dating back to 1777. Draw parallel lines on a table one inch apart. Drop a needle, also one inch long, onto this table. Simple trigonometry will show you that the probability of the needle touching one of the lines is 2/π. So conduct many such experiments to get an approximation to π. Unfortunately because of the probabilistic nature of this method you will have to drop the needle many billions of times to find πaccurate to half a dozen decimal places. There can also be a relationship between certain types of differential equation and probabilistic methods. Stanislaw Ulam, inspired by a card game, invented this technique while working on the Manhattan Project towards the development of nuclear weapons. The name ‘Monte Carlo’ was given to this idea by his colleague Nicholas Metropolis. Monte Carlo simulations are used in financial problems for solving two types of problems:• Exploring the statistical properties of a portfolio of investments or cashflows to determine quantities such as expected returns, risk, possible downsides, probabilities of making certain profits or losses, etc.• Finding the value of derivatives by exploiting the theoretical relationship between option values and expected payoff under a risk-neutral random walk.

**Exploring portfolio statistics** The most successful quantitative models represent investments as random walks. There is a whole mathematical theory behind these models, but to appreciate the role they play in portfolio analysis you just need to understand three simple concepts. First, you need an algorithm for how the most basic investments evolve randomly. In equities this is often the lognormal random walk. (If you know about the real/risk-neutral distinction then you should know that you will be using the real random walk here.) This can be represented on a spreadsheet or in code as how a stock price changes from one period to the next by adding on a random return. In the fixed-income world you may be using the BGM model to model how interest rates of various maturities evolve. In credit you may have a model that models the random bankruptcy of a company. If you have more than one such investment that you must model then you will also need to represent any interrelationships between them. This is often achieved by using correlations. Once you can perform such simulations of the basic investments then you need to have models for more complicated contracts that depend on them, these are the options/derivatives/contingent claims. For this you need some theory, derivatives theory. This the second concept you must understand. Finally, you will be able to simulate many thousands, or more, future scenarios for your portfolio and use the results to examine the statistics of this portfolio. This is, for example, how classical Value at Risk can be estimated, among other things.

**Pricing derivatives** We know from the results of risk neutral pricing that in the popular derivatives theories the value of an option can be calculated as the present value of the expected payoff under a risk-neutral random walk. And calculating expectations for a single contract is just a simple example of the above-mentioned portfolio analysis, but just for a single option and using the risk-neutral instead of the real random walk. Even though the pricing models can often be written as deterministic partial differential equations they can be solved in a probabilistic way, just as Stanislaw Ulam noted for other, non-financial, problems. This pricing methodology for derivatives was first proposed by the actuarially trained Phelim Boyle in 1977.Whether you use Monte Carlo for probabilistic or deterministic problems the method is usually quite simple to implement in basic form and so is extremely popular in practice

## Brownian Bridge



## Why do quants like closed-form solutions?

|  |  |
| --- | --- |
| **Short Answer** | |
|  | Because they are fast to compute and easy to understand. | |
|  | **Example** | |
|  | The Black–Scholes formulæ are simple and closed-form and often used despite people knowing that they have limitations, and despite being used for products for which they were not originally intended. | |
|  | **Long Answer** | |
|  | There are various pressures on a quant when it comes to choosing a model. What he’d really like is a model that is | |
|  | • **robust**: small changes in the random process for the underlying don’t matter too much | |
|  | • **fast**: prices and the greeks have to be quick to compute for several reasons, so that the trade gets done and you don’t lose out to a competitor, and so that positions can be managed in real time as just one small part of a large portfolio | |
|  | • **accurate**: in a scientific sense the prices ought to be good, perhaps matching historical data. This is different from robust, of course | |
|  | • **easy to calibrate**: banks like to have models that match traded prices of simple contracts | |
|  | There is some overlap in these. Fast may also mean easy to calibrate, but not necessarily. Accurate and robust might be similar, but again, not always. From the scientific point of view the most important of these is accuracy. The least important is speed. To the scientist the question of calibration becomes one concerning the existence of arbitrage. If you are a hedge fund looking for prop trading opportunities with vanillas then calibration is precisely what you don’t want to do. And robustness would be nice, but maybe the financial world is so unstable that models can never be robust. To the practitioner he needs to be able to price quickly to get the deal done and to manage the risk. If he is in the business of selling exotic contracts then he will invariably be calibrating, so that he can say that his prices are consistent with vanillas. As long as the model isn’t too inaccurate or sensitive, and he can add a sufficient profit margin, then he will be content. So to the practitioner speed and ability to calibrate to the market are the most important. The scientist and the practitioner have conflicting interests. And the practitioner usually wins. And what could be faster than a closed-form solution? This is why practitioners tend to favour closed forms. They also tend to be easier to understand intuitively than a numerical solution. The Black–Scholes formulæ are perfect for this, having a simple interpretation in terms of expectations, and using the cumulative distribution function for the Gaussian distribution. Such is the desire for simple formulæ that people often use the formulæ for the wrong product. Suppose you want to price certain Asian options based on an arithmetic average. To do this properly in the Black–Scholes world you would do this by solving a three-dimensional partial differential equation or by Monte Carlo simulation. But if you pretend that the averaging is geometric and not arithmetic then often there are simple closed form solutions. So use those, even though they must be wrong. The point is that they will probably be less wrong than other assumptions you are making, such as what future volatility will be. Of course, the definition of closed form is to some extent in the eye of the beholder. If an option can be priced in terms of an infinite sum of hypergeometric functions does that count? Some Asian options can be priced that way. Or what about a closed form involving a subtle integration in the complex plane that must ultimately be done numerically? That is the Heston stochastic volatility model. If closed form is so appreciated, is it worth spending much time seeking them out? Probably not. There are always new products being invented and new pricing models being devised, but they are unlikely to be of the simple type that can be solved explicitly. Chances are that either you will have to solve these numerically, or approximate them by something not too dissimilar. Approximations such as Black ’76 are probably your best chance of finding closed-form solutions for new products these days. | |

## What are the forward and backward equations?

|  |  |
| --- | --- |
| **Short Answer** | |
|  | Forward and backward equations usually refer to the differential equations governing the transition probability density function for a stochastic process. They are diffusion equations and must therefore be solved in the appropriate direction in time, hence the names. |
|  | **Example** |
|  | An exchange rate is currently 1.88. What is the probability that it will be over 2 by this time next year? If you have a stochastic differential equation model for this exchange rate then this question can be answered using the equations for the transition probability density function. |
|  | **Long Answer** |
|  | Let us suppose that we have a random variable y evolving according to a quite general, one-factor stochastic differential equation dy = A(y, t) dt + B(y, t) dX. Here A and B are both arbitrary functions of y and t. Many common models can be written in this form, including the lognormal asset random walk, and common spot interest rate models. The transition probability density function p(y, t; y , t )is the function of four variables defined by Prob(a < y < b at time t |y at time t)=\_ bap(y, t; y , t ) dy. This simply means the probability that the random variable y lies between a and b at time t in the future, given that it started out with value y at time t. You can think of y and t as being current or starting values withy and t being future values. The transition probability density function p(y, t; y , t )satisfies two equations, one involving derivatives with respect to the future state and time (y and t ) and called the forward equation, and the other involving derivatives with respect to the current state and time(y and t) and called the backward equation. These two equations are parabolic partial differential equations not dissimilar to the Black–Scholes equation. |
|  | The forward equation Also known as the Fokker–Planck or forward Kolmogorov equation |
|  | This forward parabolic partial differential equation requires initial conditions at time t and to be solved for t> t. |
|  | **Example**: An important example is that of the distribution of equity prices in the future |
|  | The backward equation Also known as the backward Kolmogorov equation this is∂p∂t+ 12B(y, t)2 ∂2p∂y2+ A(y, t)∂p∂y= 0.This must be solved backwards in t with specified final data. For example, if we wish to calculate the expected value of some function F(S) at time T we must solve this equation for the function p(S, t) with p(S, T) = F(S).Option prices If we have the lognormal random walk for S, as above, and we transform the dependent variable using a discount factor according to p(S, t) = er(T−t)V(S, t),then the backward equation for p becomes an equation for V which is identical to the Black–Scholes partial differential equation. Identical but for one subtlety, the equation contains a μ where Black–Scholes contains r. We can conclude that the fair value of an option is the present value of the expected payoff at expiration under a risk-neutral random walk for the underlying. Risk neutral here means replace μ with r. |

## What is meant by ‘complete’ and ‘incomplete’ markets?

**Short Answer**

A complete market is one in which a derivative product can be artificially made from more basic instruments, such as cash and the underlying asset. This usually involves dynamically rebalancing a portfolio of the simpler instruments, according to some formula or algorithm, to replicate the more complicated product, the derivative. Obviously, an incomplete market is one in which you can’t replicate the option with simpler instruments.

**Example**

The classic example is replicating an equity option, a call, say, by continuously buying or selling the equity so that you always hold the amount\_ = e−D(T−t)N(d1),in the stock, where N(x) = 1 √2π\_ x−∞e−12 φ2dφandd1 =ln(S/E) + (r − D + 12σ2)(T − t)σ√T – t.

**Long Answer**

A slightly more mathematical, yet still quite easily understood, description is to say that a complete market is one for which there exist the same number of linearly independent securities as there are states of the world in the future. Consider, for example, the binomial model in which there are two states of the world at the next time step, and there are also two securities, cash and the stock. That is a complete market. Now, after two time steps there will be three possible states of the world, assuming the binomial model recombines so that an up-down move gets you to the same place as down-up. You might think that you therefore need three securities for a complete market. This is not the case because after the first time step you get to change the quantity of stock you are holding, this is where the dynamic part of the replication comes in. In the equity world the two most popular models for equity prices are the lognormal, with a constant volatility, and the binomial. Both of these result in complete markets, you can replicate other contracts in these worlds. In a complete market you can replicate derivatives with the simpler instruments. But you can also turn this on its head so that you can hedge the derivative with the underlying instruments to make a risk-free instrument. In the binomial model you can replicate an option from stock and cash, or you can hedge the option with the stock to make cash. Same idea, same equations, just move terms to be on different sides of the ‘equals’ sign. As well as resulting in replication of derivatives, or the ability to hedge them, complete markets also have a nice mathematical property. Think of the binomial model. In this model you specify the probability of the stock rising (and hence falling because the probabilities must add to one). It turns out that this probability does not affect the price of the option. This is a simple consequence of complete markets, since you can hedge the option with the stock you don’t care whether the stock rises or falls, and so you don’t care what the probabilities are. People can therefore disagree on the probability of a stock rising or falling but still agree on the value of an option, as long as they share the same view on the stock’s volatility. In probabilistic terms we say that in a complete market there exists a unique martingale measure, but for an incomplete market there is no unique martingale measure. The interpretation of this is that even though options are risky instruments we don’t have to specify our own degree of risk aversion in order to price them. Enough of complete markets, where can we find incomplete markets? The answer is ‘everywhere.’ In practice, all markets are incomplete because of real-world effects that violate the assumptions of the simple models. Take volatility as an example. As long as we have a lognormal equity random walk, no transaction costs, continuous hedging, perfectly divisible assets,. . ., and constant volatility then we have a complete market. If that volatility is a known time-dependent function then the market is still complete. It is even still complete if the volatility is a known function of stock price and time. But as soon as that volatility becomes random then the market is no longer complete. This is because there are now more states of the world than there are linearly independent securities. In reality, we don’t know what volatility will be in the future so markets are incomplete. We also get incomplete markets if the underlying follows a jump-diffusion process. Again more possible states than there are underlying securities. Another common reason for getting incompleteness is if the underlying or one of the variables governing the behaviour of the underlying is random. Options on terrorist acts cannot be hedged since terrorist acts aren’t traded (to my knowledge at least).We still have to price contracts even in incomplete markets, so what can we do? There are two main ideas here. One is to price the actuarial way, the other is to try to make all option prices consistent with each other. The actuarial way is to look at pricing in some average sense. Even if you can’t hedge the risk from each option it doesn’t necessarily matter in the long run. Because in that long run you will have made many hundreds or thousands of option trades, so all that really matters is what the average price of each contract should be, even if it is risky. To some extent this relies on results from the Central Limit Theorem. This is called the actuarial approach because it is how the insurance business works. You can’t hedge the lifespan of individual policyholders but you can figure out what will happen to hundreds of thousands of them on average using actuarial tables. The other way of pricing is to make options consistent with each other. This is commonly used when we have stochastic volatility models, for example, and is also often seen in fixed-income derivatives pricing. Let’s work with the stochastic volatility model to get inspiration. Suppose we have a lognormal random walk with stochastic volatility. This means we have two sources of randomness (stock and volatility) but only one quantity with which to hedge (stock). That’s like saying that there are more states of the world than underlying securities, hence incompleteness. Well, we know we can hedge the stock price risk with the stock, leaving us with only one source of risk that we can’t get rid of. That’s like saying there is one extra degree of freedom in states of the world than there are securities .Whenever you have risk that you can’t get rid of you have to ask how that risk should be valued. The more risk the more return you expect to make in excess of the risk-free rate. This introduces the idea of the market price of risk. Technically in this case it introduces the market price of volatility risk. This measures the excess expected return in relation to unhedgeable risk. Now all options on this stock with the random volatility have the same sort of unhedgeable risk, some may have more or less risk than others but they are all exposed to volatility risk. The end result is a pricing model which explicitly contains this market price of risk parameter. This ensures that the prices of all options are consistent with each other via this ‘universal’ parameter. Another interpretation is that you price options in terms of the prices of other options.

## Interest rate modelling (Hull White, Vasicek, Ho Lee, …)

**1977 Vasicek** So far quantitative finance hadn’t had much to say about pricing interest rate products. Some people were using equity option formulæ for pricing interest rate options, but a consistent framework for interest rates had not been developed. This was addressed by Vasicek. He started by modelling a short-term interest rate as a random walk and concluded that interest rate derivatives could be valued using equations similar to the Black–Scholes partial differential equation. Oldrich Vasicek represented the short-term interest rate by a stochastic differential equation of the form dr = μ(r, t) dt + σ(r, t) dX. The bond pricing equation is a parabolic partial differential equation, similar to the Black–Scholes equation. See Vasicek (1977).

**1986 Ho and Lee** One of the problems with the Vasicek framework for interest rate derivative products was tha tit didn’t give very good prices for bonds, the simplest of fixed income products. If the model couldn’t even get bond prices right, how could it hope to correctly value bond options? Thomas Ho and Sang-Bin Lee found a way around this, introducing the idea of yield curve fitting or calibration. See Ho and Lee (1986).

**1992 Heath, Jarrow and Morton** Although Ho and Lee showed how to match theoretical and market prices for simple bonds, the methodology was rather cumbersome and not easily generalized. David Heath, Robert Jarrow and Andrew Morton took a different approach. Instead of modelling just a short rate and deducing the whole yield curve, they modelled the random evolution of the whole yield curve. The initial yield curve, and hence the value of simple interest rate instruments, was an input to the model. The model cannot easily be expressed in differential equation terms and so relies on either Monte Carlo simulation or tree building. The work was well known via a working paper, but was finally published, and therefore made respectable in Heath, Jarrow and Morton (1992).

**2002 Hagan, Kumar, Lesniewski, Woodward** There has always been a need for models that are both fast and match traded prices well. The interest-rate model of Pat Hagan, Deep Kumar, Andrew Lesniewski & Diana Woodward(2002) which has come to be called the SABR (stochastic,α, β, ρ) model is a model for a forward rate and its volatility, both of which are stochastic. This model is made tractable by exploiting an asymptotic approximation to the governing equation that is highly accurate in practice. The asymptotic analysis simplifies a problem that would otherwise have to be solved numerically. Although asymptotic analysis has been used in financial problems before, for example in modelling transaction costs, this was the first time it really entered mainstream quantitative finance.

## XVA, CVA modelling

Let us focus on a basic interest rate swap, V S(t0), as in Section 12.1.4, between two counterparties A and B. The contract is basically an exchange of Interest Rate Derivatives and fixed against floating rate payments, on a notional N. Imagine a situation in which interest rates moved substantially, so that party A will have significant financial obligations towards the counterparty B. Depending on the financial situation, it may happen that party A encounters difficulties to meet the payment obligations. Moreover, counterparty B may endure a significant financial loss due to default of party A. If the counterparty defaults, the loss will be the replacement costs of the contract (i.e. the current market value).

**Counterparty credit risk** In financial jargon the situation described above is defined as Counterparty Credit Risk.

**Definition 12.3.1 (Counterparty credit risk (CCR))** Counterparty credit risk (CCR) is related to the situation where a counterparty will default prior to the expiration of the contract and is unable make all payments required by the financial contract. In the year 2007, a financial crisis occurred, which originated in the United States’ credit and housing market, and spread around the world, from the financial markets into the real economy. Financial institutions with a high reputation went bankrupt or were bailed out, including the investment bank Lehman Brothers (founded in 1850).In the worst times of that crisis, the bankruptcy of large financial institutions triggered a widespread propagation of so-called default risk through the financial network. This initiated a thorough review of the standards and methodologies for the valuation of financial derivatives. Policies, rules and regulations in the financial world changed drastically in the wake of that crisis. An important area of financial risk which required special attention referred to the counterparty credit risk (CCR). This is the risk that a party of a financial contract is not able to fulfill the payment duties that are agreed upon in the contract, which is also known as a default. Since then, the probability of default of the counterparty of a financial contract has been incorporated in the prices of financial derivatives, and thus plays a prominent role in the pricing context. Counterparties are charged an additional premium, which is added to the fair price of the derivative, due to the probability of default. This way the risk that the counterparty would miss payment obligations is compensated for the other party in the contract. The total amount of trades of complex, and thus risky, financial derivatives has significantly reduced in the wake of the financial crisis. The lack of confidence in the financial system may have resulted in a drastic reduction of complexity, simply because risk of basic financial products is easier to estimate, and also just to keep money in the pocket. As a consequence of CCR and its effect on the value of a financial derivative, a derivative contract with a defaultable counterparty may be considered to be worthless than a contract with a free-of-risk counterparty; the lower the creditworthiness of a counterparty, the lower the market value of the derivative contract may be. From the perspective of derivatives pricing, it may be clear that pricing a derivative under the risk-neutral measure is not sufficient regarding the risk related to the possible default of a counterparty. Typically, the probability of default of a counterparty can be assessed either in an implied fashion, meaning that the relevant information is extracted from the market quotes of credit derivatives, like Credit Default Swaps (CDS), that give an indication of a companies’ or countries creditworthiness, or it can be inferred from scores and insights on the credit quality of companies and countries, from, for example, rating agencies. The pricing of a derivative under CCR is related to the exposure in the future to a specific counterparty, as this future exposure gives us an indication about the possible sizes of losses in the event of a counterparty default.

…

As a result, the CVA charge can be approximated by, CVA \_ |LG{zD} loss given default\_ |P{Dz}probability of default\_ |E{zE}expected positive exposure: In the above derivations the independence between the counterparties’ risk factors, that are modeled by 1tD\_T and the risk associated with the underlying exposure, which is governed by \_ V (tD; T), is assumed. This assumption of independence may lead an underestimation of risk under certain circumstances, which may result in a potential significant loss. The phenomenon which is then not modeled is commonly referred to as Wrong Way Risk (WWR), as compared to the so-called Right Way Risk. WWR is defined as the risk which occurs when the “exposure to a counterparty is adversely correlated with the credit quality of the counterparty”. An example of WWR is when fluctuations in the interest rate would cause changes in the value of certain derivative transactions, and also impact, at the same time, the creditworthiness of the counterparty. The counterparty is affected twice in such a situation by the specific phenomenon. Suppose a bank enters a swap contract with an oil company where the bank pays a fixed amount and receives a floating amount, and the floating amount would be linked to the oil price. When the oil price moves upwards in this example, the bank would expect to receive money from the oil producer. The producer’s credit quality, on the other hand, may also deteriorate as the costs may increase due to the price movement. Under normal circumstances, the oil producer would look for protection against a rise in the oil prices by entering a long oil derivative contract, instead of paying an amount which is linked to the oil price.

## Combinatoric questions (coin toss, dice toss etc)

1. *What is the expected number of coin flips to get a head?*
2. *Elementary probability models (cards, binomial distributions, gaussian limits of LLN)*
3. *Calculate the conditional probability; Fibonacci number; clock angle.*
4. *What is the number of coins you need to flip simultaneously to have at least a 90% chance of getting at least one head?*
5. *Person A flips a coin. If he lands heads, he wins. If he lands tails, player B flips the coin. If player B lands heads, he wins. If he lands tails, player A goes again, until somebody lands heads. What is the probability that player A will win this game?*
6. *There is a game where you roll a die. You win money for whatever you roll ($1 if you roll a 1, $2 if you roll a 2, etc). If you are unhappy with your first roll, you are able to roll the die a second time. If you roll the die a second time, only your second roll determines the payout. What is a fair price to charge to play this game?*
7. *St. Petersburg paradox*
8. *Compute the expected value of exp(X) with X ~N(0,1).*
9. *The expected times of coin tosses to get two consecutive heads.*
10. *There are 3 boxes. Each box includes 2 balls. The first one includes 2 black ball, the second one includes one black and one white and the last one includes 2 white balls. Pick 2 balls from one box. The first one is white. Then, what's the probability that the second one is also be a white?*
11. *A dice is rolled twice, what is the probability that on the second chance it would be 6.*
12. *What's the prob of getting two equal numbers if you roll two dices*
13. *7 fair coins 1 double headed coin. I pick a coin and toss it three times. I get heads all three times. What is the probability it is the double headed coin?*
14. *Statistics problems: binomial distribution and law of large number.*
15. *One is a probability question about poker cards*
16. *Optimization problems using Lagrange optimizer.*

## Links

[*https://www.quantstart.com/articles*](https://www.quantstart.com/articles)

*Willmott, FAQ in Quant Finance*

*https://en.wikipedia.org/wiki/Greeks\_(finance)*

[*https://www.optionsplaybook.com/options-introduction/option-greeks/*](https://www.optionsplaybook.com/options-introduction/option-greeks/)

*https://www.treasurer.ca.gov/cdiac/publications/math.pdf*

[*http://www.ict.nsc.ru/jct/getfile.php?id=461*](http://www.ict.nsc.ru/jct/getfile.php?id=461)

*https://en.wikipedia.org/wiki/Volatility\_smile*

[*https://blog.quantinsti.com/volatility-smile-origin-implications/*](https://blog.quantinsti.com/volatility-smile-origin-implications/)

*https://en.wikipedia.org/wiki/Stochastic\_volatility*

[*https://www.researchgate.net/publication/239059413\_Riding\_on\_a\_Smile*](https://www.researchgate.net/publication/239059413_Riding_on_a_Smile)

[*https://math.stackexchange.com/questions/1273437/brownian-motion-and-covariance*](https://math.stackexchange.com/questions/1273437/brownian-motion-and-covariance)

[*https://quant.stackexchange.com/questions/341/what-is-a-martingale*](https://quant.stackexchange.com/questions/341/what-is-a-martingale)

<https://ethz.ch/content/dam/ethz/special-interest/mavt/dynamic-systems-n-control/idsc-dam/Lectures/Stochastic-Systems/SDE.pdf>

[*https://www.albany.edu/~bd445/Economics\_802\_Financial\_Economics\_Slides\_Fall\_2013/Black-Scholes\_Option\_Pricing.pdf*](https://www.albany.edu/~bd445/Economics_802_Financial_Economics_Slides_Fall_2013/Black-Scholes_Option_Pricing.pdf)

[*https://www.cantorsparadise.com/the-black-scholes-formula-explained-9e05b7865d8a*](https://www.cantorsparadise.com/the-black-scholes-formula-explained-9e05b7865d8a)

[*https://papers.ssrn.com/sol3/papers.cfm?abstract\_id=3756731*](https://papers.ssrn.com/sol3/papers.cfm?abstract_id=3756731)

[*https://viewpoint.pwc.com/dt/us/en/pwc/accounting\_guides/stockbased\_compensat/stockbased\_compensat\_\_3\_US/chapter\_8\_estimating\_US/84\_the\_blackscholes\_\_US.html*](https://viewpoint.pwc.com/dt/us/en/pwc/accounting_guides/stockbased_compensat/stockbased_compensat__3_US/chapter_8_estimating_US/84_the_blackscholes__US.html)

[*http://www.columbia.edu/~mh2078/ContinuousFE/BlackScholesCtsTime.pdf*](http://www.columbia.edu/~mh2078/ContinuousFE/BlackScholesCtsTime.pdf)

[*https://www.cantorsparadise.com/the-black-scholes-formula-explained-9e05b7865d8a*](https://www.cantorsparadise.com/the-black-scholes-formula-explained-9e05b7865d8a)

[*https://en.wikipedia.org/wiki/Volatility\_smile*](https://en.wikipedia.org/wiki/Volatility_smile)

[*https://blog.quantinsti.com/volatility-smile-origin-implications/*](https://blog.quantinsti.com/volatility-smile-origin-implications/)

# ****Econometrics / ML****

## OLS

## Logistic

## ARMA/ARIMA/SARIMA/ARMAX

## ARDL

## Mean-variance trade-off

## Training / testing sample

## Classification measures (recall, sensitivity, F1 score, confusion matrix)

## VAR

## ARCH / GARCH

## Cointergration

## Kalman Filtering

## Signal Processing

## Random Forrest

## XGBoost

## PCA

# ****Balance sheet modelling / ALM****

## Balance Sheet optimization

The set of actions / models, whose target is to determine asset allocation such that bank’s ROE is maximized given the capital constraints (regulatory requirements, strategic objectives etc)

## ALM behavioral model

Client behavior deeply impacts a bank's liquidity, funding, interest-rate position and, consequently, the management of its asset/liability mismatch and related profitability. Therefore it is imperative that risk managers and modelers alike understand how to model client behavior according to the needs of their business. It is the aim of this book to improve that understanding and highlight modelling techniques from the simple through to the complex, offering a broad suite of tools to improve the management of an institution's balance sheet.

## Economic life and stability of Non-Maturing Deposits

Non-maturity deposits (‘NMDs’), such as retail savings, interest and non-interest-bearing checking and money-market accounts have no stated maturities. Thus, depositors can withdraw their funds at any time without any penalty. Modeling this early redemption option can be very challenging, but very rewarding, since NMDs are typically viewed as one of the most stable sources of funding for banks’ assets. Banks have other means of funding at their disposal, such as FHLB advances or the repo market but holding a substantial proportion of core deposits allows easy access to a stable and cheaper source of funding. Due to their short-term maturity and repricing nature, these instruments have relatively low interest rate risk which makes them more attractive.

## Early redemptions of term deposits / prepayments / embedded optionality

Risk for the bank stating that the customers may withdraw their funds early (liability side)

# ****Portfolio Theory / Trading****

## Markowitz

## CAPM

## Mean- Variance portfolio optim

## Pairs Trading

## Backtesting strategies

## High Frequency Trading

## Factor investing

# ****Banking****

## What is banking?

## Investment Banking

## Retail Loans

## Wholesale Loans

## Credit Cards

## Trading

## Asset & Wealth Management – Private Banking

## Mortgages

## Banking Book vs Trading Book

## NIR

## NII

Non-interest income is bank and creditor income derived primarily from fees including deposit and transaction fees, insufficient funds (NSF) fees, annual fees, monthly account service charges, inactivity fees, check and deposit slip fees, and so on. Credit card issuers also charge penalty fees, including late fees and over-the-limit fees. Institutions charge fees that generate non-interest income as a way of increasing revenue and ensuring liquidity in the event of increased default rates.

## NIE

## Charge-Offs / Provisions

## Bank Balance Sheet

## RWA

# ****Financial instruments****

## Deposits

## Fixed Income (Sovereign bonds, corporate bonds)

## Repo/reversed repo

## IRS

## Interbank rates (IBOR, SOFR etc)

## Swaptions

***Q: Do you need to know volatility surface to price the variable leg of a swap?***

**A:** A volatility surface has to be fit using a number of derivatives, it’s not specific to one swap.

While there are some technical differences in precise definition, the general idea is that the surface tells you the implied volatility of the reference interest rate at any future time and level of interest rates.

It might be easier to understand if you consider simulating an interest rate path. Suppose three-month LIBOR is 1.00% now, and at t=0, r = 1% it’s daily volatility is 0.03%. So you would simulate a draw from a distribution with that volatility and get, say, 0.98% for tomorrow. Looking at the volatility surface for t=1, r=0.98%, you’d get a new volatility, to simulate the next move. And so on into the future.

## Equities

## Forward/Futures

## Equity derivatives

## FX derivatives

## Commodities

## Commodity derivatives

## Crypto ???

## Asset Backed Securities

## Mortgage Backed Securities

# ****Stress testing****

## FRTB

## Basel

Basel III accord was scheduled to be implemented effective March 2019. In view of the coronavirus pandemic, the implementation has been postponed by a year till January 1, 2023. Basel III has incorporated several risk measures to counter issues which were identified and highlighted in 2008 financial crisis. It emphasis on revised capital standards (such as leverage ratios), stress testing and tangible equity capital which is the component with the greatest loss-absorbing capacity.

The concept of building internal models and external ratings for estimating PD, LGD and EAD remains same as it was in Basel II. However there are some changes introduced in Basel III. It is shown in the table below.

|  |  |  |
| --- | --- | --- |
|  | Basel II | Basel III |
| Common Tier 1 capital ratio(shareholders’ equity + retained earnings) | 2% \* RWA | 4.5% \* RWA |
| Tier 1 capital ratio | 4% \* RWA | 6% \* RWA |
| Tier 2 capital ratio | 4% \* RWA | 2% \* RWA |
| Capital conservation buffer(common equity) | - | 2.5% \* RWA |

## CCAR

## ICAAP

## IFRS9

IFRS 9 is an International Financial Reporting Standard dealing with accounting for financial instruments. It replaces IAS 39 Financial Instruments which was based on the incurred loss model whereas IFRS 9 focuses on the expected loss model that covers also future losses.

In IFRS 9, the idea is to recognize 12-month loss allowance at initial recognition and lifetime loss allowance on significant increase in credit risk

As per IFRS 9, there are three stages of Credit Risk which are as follows -

1. Stage 1 - Credit risk has not increased significantly since initial recognition, indicates low credit risk at reporting date
2. Stage 2 - Credit risk has increased significantly since initial recognition
3. Stage 3 - Permanent reduction in the value of financial asset at the reporting date

### How IFRS 9 is different from Basel III?

Yes, they are different but both requires building PD, LGD and EAD models. See the difference between them below.

|  |  |  |
| --- | --- | --- |
| **Parameters** | **Basel III** | **IFRS 9** |
| **Objective** | Expected + Unexpected Loss | Expected Loss |
| **PD** | One year PD | 12 month PD for stage 1 assets, Lifetime PD for stage 2 and 3 assets |
| **Rating Philosophy** | TTC rating philosophy | PIT rating philosophy |
| **LGD** | Downturn LGD (both direct + indirect costs) | Best estimate LGD (only direct costs) |
| **EAD** | Downturn EAD | Best estimate EAD |
| **Expected Loss /Expected CreditLoss (ECL)** | EL=PD\*LGD\*EAD | EL=PD\*PV of cash shortfalls |

## Tier 1

## Tier 2

## CET1

## RWA

The capital adequacy ratio is calculated as eligible capital divided by risk-weighted assets.

Risk-weighted assets, or RWA, are used to link the minimum amount of capital that banks must have, with the risk profile of the bank’s lending activities (and other assets). The more risk a bank is taking, the more capital is needed to protect depositors.

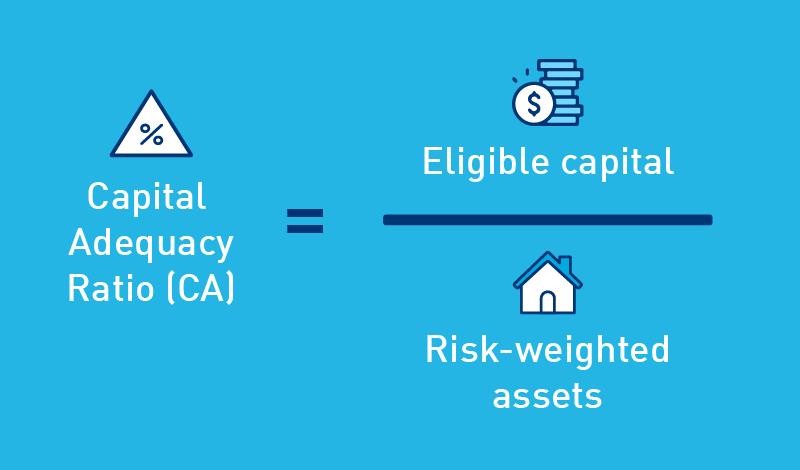
Essentially, risk-weighted assets are the loans and other assets of a bank, weighted (that is, multiplied by a percentage factor) to reflect their respective level of risk of loss to the bank. For example, mortgages secured by residential property are generally considered to be lower risk than unsecured credit card lending. As such, the greater the amount of higher risk assets and loans that a bank has, the higher its risk-weighted assets, and therefore, the higher the amount of capital the bank must have in order to meet APRA’s minimum capital adequacy ratios.

Risk-weighted assets are designed to be responsive to changes in both the quality and composition of a bank’s lending portfolio. In an economic downturn, as credit risk increases, risk-weighted assets would also be expected to increase to some extent.

There are two broad approaches to calculating credit risk-weighted assets:

1. The standardised approach, and
2. The internal ratings-based (IRB) approach.

Typically, only the larger banks have the expertise and infrastructure necessary to use the IRB approach. However, under both approaches, risk-weighted assets are broadly calculated as the risk weight multiplied by the exposure amount.



Risk weights are essentially percentage factors that adjust for the credit risk of different types of assets. Under the standardised approach, banks must apply APRA-prescribed risk weights. For example, cash and Australian Government securities are considered to represent a lower risk (zero per cent risk weight) than an unsecured loan to a business (100 per cent risk weight). Under the IRB approach, banks are allowed to use APRA-approved models to calculate risk estimates (for example, the probability of default) to assess the credit risk of borrowers. These risk estimates are then used as inputs in APRA-prescribed formulas.

### Exposure amount

The exposure amount refers to the total amount that a bank could potentially lose if a borrower defaults. This includes not only the funds actually advanced by a bank (the drawn or on-balance sheet amount), but also any additional amounts up to advised limits that the bank may be contractually obliged to advance in the future (undrawn or off-balance sheet amount).

To calculate the on-balance sheet equivalent for off-balance sheet amounts, a bank multiplies the undrawn amount by a credit conversion factor (which represents the likelihood of a bank being required to advance additional funds).

### Calculating risk-weighted assets

Banks calculate risk-weighted assets by multiplying the exposure amount by the relevant risk weight for the type of loan or asset. A bank repeats this calculation for all of its loans and assets, and adds them together to calculate total credit risk-weighted assets.

The example below assumes that a bank has total assets worth $100:

| Asset/Loan | Amount ($) | Risk weight (%) | RWA ($) |
| --- | --- | --- | --- |
| **Cash** | $20 | 0% | $0 |
| **Australian Government securities** | $10 | 0% | $0 |
| **Housing loans** | $40 | 35% | $14 |
| **Business loans** | $30 | 100% | $30 |
| **Total** | $100 | N/A | $44 |

The total credit risk-weighted assets ($44 in the example above) are added to risk-weighted assets for operational risk and market risk, and used as the denominator in the capital adequacy ratio.

The higher a bank’s total risk-weighted assets, the more capital it will need in order to meet the minimum capital adequacy ratios set by APRA. For example:

Assume a bank has $10 of capital and seeks to have a 10% capital ratio. Then the bank can have $100 of risk-weighted assets. (Figure 1)

If the bank’s risk-weighted assets grow to $120, then its capital ratio will fall to 8.3%. (Figure 2)

So if the bank wishes to maintain a 10% capital ratio, it will need to raise an extra $2 of capital. (Figure 3)

|  |  |  |
| --- | --- | --- |
| Figure 3  Figure 1 | Figure 2  Figure 2 | Figure 1  Figure 3 |

The risk-weighted assets calculation therefore plays an important role in ensuring that the capital held by banks corresponds with their overall risk profile.

## Leverage Ratio

## Links

* CCAR

<https://www.risk.net/definition/comprehensive-capital-analysis-and-review-ccar>

<https://www.federalreserve.gov/publications/files/2022-dfast-results-20220623.pdf>

* IFRS 9

<https://www.pwc.com/gx/en/audit-services/ifrs/publications/ifrs-9/ifrs-9-understanding-the-basics.pdf>

* CECL

<https://en.wikipedia.org/wiki/Current_Expected_Credit_Losses>

* ICAAP

<https://home.kpmg/md/en/home/services/advisory/consulting/risk-compliance-services/integrated-risk-management/icaap--economic-capital-management.html>

* **RWA calculations**

<https://www.apra.gov.au/apra-explains-risk-weighted-assets>

<https://www.wallstreetmojo.com/risk-weighted-asset/>

<https://www.bis.org/publ/bcbs128b.pdf>

# ****Market Risk****

## Economic Capital

## EVE

The economic value of equity (EVE) is a cash flow calculation that takes the present value of all asset cash flows and subtracts the present value of all liability cash flows. Unlike earnings at risk and value at risk (VAR), a bank uses the economic value of equity to manage its assets and liabilities. This is a long-term economic measure used to assess the degree of interest rate risk exposure—as opposed to net-interest income (NII), which reflects short-term interest rate risk.

The simplest definition of EVE is the net present value (NPV) of a bank's balance sheet's cash flows. This calculation is used for asset-liability management to measure changes in the economic value of the bank.

## VaR

**Three ways to estimate VaR:**

1. Variance-covariance matrices

2. Historical simulation

3. Monte Carlo simulation

**Shortcomings of VaR:**

* Lack of subadditivity of risks, even independent risks, which creates aggregation problems (i.e., VaR of a portfolio with two instrument may be greater than the sum of the independent VaRs of these instruments).
* VaR does not encourage and, indeed, sometimes prohibits diversification.

## ES

## SVaR

## CVaR

## Consistent Risk Measure

## Greeks

## European / Vanilla Equity Options (valuation)

## Exotic Equity Options (valuation)

* American
* Bermudian
* Asian
* Binary
* Chooser Option
* Call on Call
* Barrier options
* Lookback options
* Basket / Multi-asset options (Curse of Dimensionality)

## Options strategies (valuation)

* Up/down play
* Collar spread
* Synthetic options
* Stellar
* Bull/bear butterfly

## Interest rate derivatives (valuation)

## FX derivatives (valuation)

## Commodities derivatives (valuation)

# ****Credit Risk****

## PD

**Probability of default** means the likelihood that a borrower will default on debt (credit card, mortgage or non-mortgage loan) over a one-year period. In simple words, it returns the expected probability of customers fail to repay the loan. Probability is expressed in the form of percentage, lies between 0% and 100%. Higher the probability, higher the chance of default.

## LGD

It means how much of the amount outstanding we expect to lose. It is a proportion of the total exposure when borrower defaults. It is calculated by (1 - Recovery Rate).

*LGD = (EAD – PV(recovery) – PV(cost)) / EAD*

*PV (recovery)= Present value of recovery discounted till time of default.*

*PV (cost) = Present value of cost discounted till time of default.*

## EAD

It means how much should we expect the amount outstanding to be in the case of default. It is the amount that the borrower has to pay the bank at the time of default.

## Modelling

## Links

<https://www.listendata.com/2019/08/credit-risk-modelling.html>

<https://www.investopedia.com/terms/l/lossgivendefault.asp>

<https://www.cambridge.org/core/books/abs/managing-portfolio-credit-risk-in-banks/exposure-at-default-ead-and-loss-given-default-lgd/16CF4DBFCB8940BA893E08CDD2156424>

<https://www.imsl.com/blog/credit-risk-modeling>

<https://www.bankofengland.co.uk/-/media/boe/files/ccbs/resources/modelling-credit-risk.pdf>

**!!! WRONG WAY RISK**

<https://www.youtube.com/watch?v=winQg35BhkU>

**!!! Example of CVA calculation**

<https://www.youtube.com/watch?v=vKBV7YCQi1Y>

# ****Liquidity Risk****

## LCR

The LCR (Liquidity Capital Ratio) is an essential component of the Basel III reforms, which are global regulatory standards on bank capital adequacy and liquidity endorsed by the G20 Leaders.

The LCR promotes the short-term resilience of a bank's liquidity risk profile. It does this by ensuring that a bank has an adequate stock of unencumbered high-quality liquid assets (HQLA) that can be converted into cash easily and immediately in private markets to meet its liquidity needs for a 30 calendar day liquidity stress scenario. It will improve the banking sector's ability to absorb shocks arising from financial and economic stress, whatever the source, thus reducing the risk of spillover from the financial sector to the real economy.

## NSFR (Net Stable Funding Ratio)

The NSFR is expressed as a ratio that must equal or exceed 100%. The ratio relates the bank's available stable funding to its required stable funding, as summarised in the following formula:

https://www.bis.org/fsi/fsisummaries/nsfr_formula.jpg

To determine total ASF and RSF amounts, factors reflecting supervisory assumptions are assigned to the bank's sources of funding and to its exposures, with these factors reflecting the liquidity characteristics of each category of instruments.

## Liquidity cashflow mismatch (LMI)

Their "Liquidity Mismatch Index" (LMI) measures the mismatch between the market liquidity of assets and the funding liquidity of liabilities, at a firm level. There are many empirical challenges that arise in implementing their theoretical measure

## Deposit concentration

A risk situation, in which a large part of the deposits sits in only a few institutions

## Liquidity stress testing

Liquidity stress testing: Considers a financial institution's ability, in the absence of market or funding liquidity, to meet obligations during periods of stress by accurately measuring the liquidity profile of the balance sheet at an enterprise-wide level.

## FTP

The Fund Transfer Pricing (FTP) measures the contribution by each source of funding to the overall profitability in a financial institution. Funds that go toward lending products are charged to asset-generating businesses whereas funds generated by deposit and other funding products are credited to liability-generating businesses.

# ****Treasury / Interest Rate Risk:****

## Interest Rate Risk

Interest rate risk in the banking book (IRRBB) refers to the current or prospective risk to the bank’s capital and earnings arising from adverse movements in interest rates that affect the bank’s banking book positions. When interest rates change, the present value and timing of future cash flows change. This in turn changes the underlying value of a bank’s assets, liabilities and off-balance sheet items and hence its economic value. Changes in interest rates also affect a bank’s earnings by altering interest rate-sensitive income and expenses, affecting its net interest income (NII). Excessive IRRBB can pose a significant threat to a bank’s current capital base and/or future earnings if not managed appropriately. A more detailed description of IRRBB and its management techniques can be found in SRP98.

31.2

Three main sub-types of IRRBB are defined for the purposes of this chapter. All three sub-types of IRRBB potentially change the price/value or earnings/costs of interest rate-sensitive assets, liabilities and/or off-balance sheet items in a way, or at a time, that can adversely affect a bank’s financial condition.

(1)

Gap risk arises from the term structure of banking book instruments, and describes the risk arising from the timing of instruments’ rate changes. The extent of gap risk depends on whether changes to the term structure of interest rates occur consistently across the yield curve (parallel risk) or differentially by period (non-parallel risk).

(2)

Basis risk describes the impact of relative changes in interest rates for financial instruments that have similar tenors but are priced using different interest rate indices.

(3)

Option risk arises from option derivative positions or from optional elements embedded in a bank’s assets, liabilities and/or off-balance sheet items, where the bank or its customer can alter the level and timing of their cash flows. Option risk can be further characterised into automatic option risk and behavioural option risk.

While the three sub-types listed above are directly linked to IRRBB, credit spread risk in the banking book (CSRBB) is a related risk that banks need to monitor and assess in their interest rate risk management framework. CSRBB refers to any kind of asset/liability spread risk of credit-risky instruments that is not explained by IRRBB and by the expected credit/jump to default risk

## PV01

Gap analysis can be used to derive the duration profile of the banking book or, equivalently, **the profile of the present value of a single basis point change in interest rates (PV01).** Gap analysis allocates all relevant interest rate-sensitive assets and liabilities to a certain number of predefined time buckets according to their next contractual reset date. The analysis also allocates equity, NMDs, prepaying loans or other instruments with future cash flows subject to customer behaviors according to general/behavioral assumptions regarding their maturity or reset date. It then measures the arithmetic difference (the gap) between the amounts of assets and liabilities in each time bucket, in absolute terms.

Each time bucket gap can be multiplied by an assumed change in interest rates to yield an approximation of the change in NII that would result from an increase in interest rates. This method gives a visual impression of the risk exposure dispersion relative to the repricing profile, reflecting exposures to parallel as well as non-parallel gap risk. It does not, however, quantify this risk.40 The measure assumes that all positions within a particular time bucket mature and reprice simultaneously, ignoring potential basis risks within the gaps.

## Duration matching

## Yield curve

## Funding costs

# ****Counterparty Risk****

# ****Commodity Risk****

# ****FX Risk****

# ****Python****

## Data structures like arrays/DataFrame/dictionary/tuple/list

* **More on Lists**

**list.append(x)**

Add an item to the end of the list. Equivalent to a[len(a):] = [x].

**list.extend(iterable)­­**

Extend the list by appending all the items from the iterable. Equivalent to a[len(a):] = iterable.

**list.insert(i, x)**

Insert an item at a given position. The first argument is the index of the element before which to insert, so a.insert(0, x) inserts at the front of the list, and a.insert(len(a), x) is equivalent to a.append(x).

**list.remove(x)**

Remove the first item from the list whose value is equal to x. It raises a ValueError if there is no such item.

**list.pop([i])**

Remove the item at the given position in the list, and return it. If no index is specified, a.pop() removes and returns the last item in the list. (The square brackets around the i in the method signature denote that the parameter is optional, not that you should type square brackets at that position. You will see this notation frequently in the Python Library Reference.)

**list.clear()**

Remove all items from the list. Equivalent to del a[:].

**list.index(x[, start[, end]])**

Return zero-based index in the list of the first item whose value is equal to x. Raises a ValueError if there is no such item.

The optional arguments start and end are interpreted as in the slice notation and are used to limit the search to a particular subsequence of the list. The returned index is computed relative to the beginning of the full sequence rather than the start argument.

**list.count(x)**

Return the number of times x appears in the list.

**list.sort(\*, key=None, reverse=False)**

Sort the items of the list in place (the arguments can be used for sort customization, see sorted() for their explanation).

**list.reverse()**

Reverse the elements of the list in place.

**list.copy()**

Return a shallow copy of the list. Equivalent to a[:].

* **Using Lists as Stacks**

The list methods make it very easy to use a list as a stack, where the last element added is the first element retrieved (“last-in, first-out”). To add an item to the top of the stack, use append(). To retrieve an item from the top of the stack, use pop() without an explicit index. For example:

* **>>>** stack = [3, 4, 5]
* **>>>** stack.append(6)
* **>>>** stack.append(7)
* **>>>** stack
* [3, 4, 5, 6, 7]
* **>>>** stack.pop()
* 7
* **>>>** stack
* [3, 4, 5, 6]
* **>>>** stack.pop()
* 6
* **>>>** stack.pop()
* 5
* **>>>** stack
* [3, 4]
* *Use a queue when you want to get things out in the order that you put them in.*
* *Use a stack when you want to get things out in the reverse order than you put them in.*
* *Use a list when you want to get anything out, regardless of when you put them in (and when you don't want them to automatically be removed).*
* Using Lists as Queues

It is also possible to use a list as a queue, where the first element added is the first element retrieved (“first-in, first-out”); however, lists are not efficient for this purpose. While appends and pops from the end of list are fast, doing inserts or pops from the beginning of a list is slow (because all of the other elements have to be shifted by one).

To implement a queue, use [collections.deque](https://docs.python.org/3/library/collections.html#collections.deque) which was designed to have fast appends and pops from both ends. For example:

>>>

**>>> from** **collections** **import** deque

**>>>** queue = deque(["Eric", "John", "Michael"])

**>>>** queue.append("Terry") *# Terry arrives*

**>>>** queue.append("Graham") *# Graham arrives*

**>>>** queue.popleft() *# The first to arrive now leaves*

'Eric'

**>>>** queue.popleft() *# The second to arrive now leaves*

'John'

**>>>** queue *# Remaining queue in order of arrival*

deque(['Michael', 'Terry', 'Graham'])

* **List Comprehensions**

List comprehensions provide a concise way to create lists. Common applications are to make new lists where each element is the result of some operations applied to each member of another sequence or iterable, or to create a subsequence of those elements that satisfy a certain condition.

For example, assume we want to create a list of squares, like:

* **>>>** squares = []
* **>>> for** x **in** range(10):
* **...**  squares.append(x\*\*2)
* **...**
* **>>>** squares
* [0, 1, 4, 9, 16, 25, 36, 49, 64, 81]

Note that this creates (or overwrites) a variable named x that still exists after the loop completes. We can calculate the list of squares without any side effects using:

squares = list(map(**lambda** x: x\*\*2, range(10)))

or, equivalently:

squares = [x\*\*2 **for** x **in** range(10)]

which is more concise and readable.

A list comprehension consists of brackets containing an expression followed by a for clause, then zero or more for or if clauses. The result will be a new list resulting from evaluating the expression in the context of the for and if clauses which follow it. For example, this listcomp combines the elements of two lists if they are not equal:

**>>>** [(x, y) **for** x **in** [1,2,3] **for** y **in** [3,1,4] **if** x != y]

[(1, 3), (1, 4), (2, 3), (2, 1), (2, 4), (3, 1), (3, 4)]

and it’s equivalent to:

**>>>** combs = []

**>>> for** x **in** [1,2,3]:

**...**  **for** y **in** [3,1,4]:

**...**  **if** x != y:

**...**  combs.append((x, y))

**...**

**>>>** combs

[(1, 3), (1, 4), (2, 3), (2, 1), (2, 4), (3, 1), (3, 4)]

Note how the order of the [for](https://docs.python.org/3/reference/compound_stmts.html#for) and [if](https://docs.python.org/3/reference/compound_stmts.html#if) statements is the same in both these snippets.

If the expression is a tuple (e.g. the (x, y) in the previous example), it must be parenthesized.

**>>>** vec = [-4, -2, 0, 2, 4]

**>>>** *# create a new list with the values doubled*

**>>>** [x\*2 **for** x **in** vec]

[-8, -4, 0, 4, 8]

**>>>** *# filter the list to exclude negative numbers*

**>>>** [x **for** x **in** vec **if** x >= 0]

[0, 2, 4]

**>>>** *# apply a function to all the elements*

**>>>** [abs(x) **for** x **in** vec]

[4, 2, 0, 2, 4]

**>>>** *# call a method on each element*

**>>>** freshfruit = [' banana', ' loganberry ', 'passion fruit ']

**>>>** [weapon.strip() **for** weapon **in** freshfruit]

['banana', 'loganberry', 'passion fruit']

**>>>** *# create a list of 2-tuples like (number, square)*

**>>>** [(x, x\*\*2) **for** x **in** range(6)]

[(0, 0), (1, 1), (2, 4), (3, 9), (4, 16), (5, 25)]

**>>>** *# the tuple must be parenthesized, otherwise an error is raised*

**>>>** [x, x\*\*2 **for** x **in** range(6)]

File "<stdin>", line 1

[x, x\*\*2 **for** x **in** range(6)]

^^^^^^^

SyntaxError: did you forget parentheses around the comprehension target?

**>>>** *# flatten a list using a listcomp with two 'for'*

**>>>** vec = [[1,2,3], [4,5,6], [7,8,9]]

**>>>** [num **for** elem **in** vec **for** num **in** elem]

[1, 2, 3, 4, 5, 6, 7, 8, 9]

List comprehensions can contain complex expressions and nested functions:

**>>> from** **math** **import** pi

**>>>** [str(round(pi, i)) **for** i **in** range(1, 6)]

['3.1', '3.14', '3.142', '3.1416', '3.14159']

* **Nested List Comprehensions**

The initial expression in a list comprehension can be any arbitrary expression, including another list comprehension.

Consider the following example of a 3x4 matrix implemented as a list of 3 lists of length 4:

>>>

**>>>** matrix = [

**...**  [1, 2, 3, 4],

**...**  [5, 6, 7, 8],

**...**  [9, 10, 11, 12],

**...** ]

The following list comprehension will transpose rows and columns:

>>>

**>>>** [[row[i] **for** row **in** matrix] **for** i **in** range(4)]

[[1, 5, 9], [2, 6, 10], [3, 7, 11], [4, 8, 12]]

As we saw in the previous section, the inner list comprehension is evaluated in the context of the [for](https://docs.python.org/3/reference/compound_stmts.html#for) that follows it, so this example is equivalent to:

>>>

**>>>** transposed = []

**>>> for** i **in** range(4):

**...**  transposed.append([row[i] **for** row **in** matrix])

**...**

**>>>** transposed

[[1, 5, 9], [2, 6, 10], [3, 7, 11], [4, 8, 12]]

which, in turn, is the same as:

>>>

**>>>** transposed = []

**>>> for** i **in** range(4):

**...**  *# the following 3 lines implement the nested listcomp*

**...**  transposed\_row = []

**...**  **for** row **in** matrix:

**...**  transposed\_row.append(row[i])

**...**  transposed.append(transposed\_row)

**...**

**>>>** transposed

[[1, 5, 9], [2, 6, 10], [3, 7, 11], [4, 8, 12]]

In the real world, you should prefer built-in functions to complex flow statements. The [zip()](https://docs.python.org/3/library/functions.html#zip) function would do a great job for this use case:

>>>

**>>>** list(zip(\*matrix))

[(1, 5, 9), (2, 6, 10), (3, 7, 11), (4, 8, 12)]

See [Unpacking Argument Lists](https://docs.python.org/3/tutorial/controlflow.html#tut-unpacking-arguments) for details on the asterisk in this line.

* **The del statement**
* There is a way to remove an item from a list given its index instead of its value: the [del](https://docs.python.org/3/reference/simple_stmts.html#del) statement. This differs from the pop() method which returns a value. The del statement can also be used to remove slices from a list or clear the entire list (which we did earlier by assignment of an empty list to the slice). For example:
* >>>
* **>>>** a = [-1, 1, 66.25, 333, 333, 1234.5]
* **>>> del** a[0]
* **>>>** a
* [1, 66.25, 333, 333, 1234.5]
* **>>> del** a[2:4]
* **>>>** a
* [1, 66.25, 1234.5]
* **>>> del** a[:]
* **>>>** a
* []
* [del](https://docs.python.org/3/reference/simple_stmts.html#del) can also be used to delete entire variables:
* >>>
* **>>> del** a
* Referencing the name a hereafter is an error (at least until another value is assigned to it). We’ll find other uses for [del](https://docs.python.org/3/reference/simple_stmts.html#del) later.
* **Tuples and Sequences**

We saw that lists and strings have many common properties, such as indexing and slicing operations. They are two examples of *sequence* data types (see [Sequence Types — list, tuple, range](https://docs.python.org/3/library/stdtypes.html#typesseq)). Since Python is an evolving language, other sequence data types may be added. There is also another standard sequence data type: the *tuple*.

A tuple consists of a number of values separated by commas, for instance:

>>>

**>>>** t = 12345, 54321, 'hello!'

**>>>** t[0]

12345

**>>>** t

(12345, 54321, 'hello!')

**>>>** *# Tuples may be nested:*

**...** u = t, (1, 2, 3, 4, 5)

**>>>** u

((12345, 54321, 'hello!'), (1, 2, 3, 4, 5))

**>>>** *# Tuples are immutable:*

**...** t[0] = 88888

Traceback (most recent call last):

File "<stdin>", line 1, in <module>

TypeError: 'tuple' object does not support item assignment

**>>>** *# but they can contain mutable objects:*

**...** v = ([1, 2, 3], [3, 2, 1])

**>>>** v

([1, 2, 3], [3, 2, 1])

As you see, on output tuples are always enclosed in parentheses, so that nested tuples are interpreted correctly; they may be input with or without surrounding parentheses, although often parentheses are necessary anyway (if the tuple is part of a larger expression). It is not possible to assign to the individual items of a tuple, however it is possible to create tuples which contain mutable objects, such as lists.

Though tuples may seem similar to lists, they are often used in different situations and for different purposes. Tuples are [immutable](https://docs.python.org/3/glossary.html#term-immutable), and usually contain a heterogeneous sequence of elements that are accessed via unpacking (see later in this section) or indexing (or even by attribute in the case of [namedtuples](https://docs.python.org/3/library/collections.html#collections.namedtuple)). Lists are [mutable](https://docs.python.org/3/glossary.html#term-mutable), and their elements are usually homogeneous and are accessed by iterating over the list.

A special problem is the construction of tuples containing 0 or 1 items: the syntax has some extra quirks to accommodate these. Empty tuples are constructed by an empty pair of parentheses; a tuple with one item is constructed by following a value with a comma (it is not sufficient to enclose a single value in parentheses). Ugly, but effective. For example:

>>>

**>>>** empty = ()

**>>>** singleton = 'hello', *# <-- note trailing comma*

**>>>** len(empty)

0

**>>>** len(singleton)

1

**>>>** singleton

('hello',)

The statement t = 12345, 54321, 'hello!' is an example of *tuple packing*: the values 12345, 54321 and 'hello!' are packed together in a tuple. The reverse operation is also possible:

>>>

**>>>** x, y, z = t

This is called, appropriately enough, *sequence unpacking* and works for any sequence on the right-hand side. Sequence unpacking requires that there are as many variables on the left side of the equals sign as there are elements in the sequence. Note that multiple assignment is really just a combination of tuple packing and sequence unpacking.

* **Sets**

Python also includes a data type for *sets*. A set is an **unordered** collection with no duplicate elements. Basic uses include membership testing and eliminating duplicate entries. Set objects also support mathematical operations like union, intersection, difference, and symmetric difference.

Curly braces or the [set()](https://docs.python.org/3/library/stdtypes.html#set) function can be used to create sets. Note: to create an empty set you have to use set(), not {}; the latter creates an empty dictionary, a data structure that we discuss in the next section.

Here is a brief demonstration:

>>>

**>>>** basket = {'apple', 'orange', 'apple', 'pear', 'orange', 'banana'}

**>>>** print(basket) *# show that duplicates have been removed*

{'orange', 'banana', 'pear', 'apple'}

**>>>** 'orange' **in** basket *# fast membership testing*

True

**>>>** 'crabgrass' **in** basket

False

**>>>** *# Demonstrate set operations on unique letters from two words*

**...**

**>>>** a = set('abracadabra')

**>>>** b = set('alacazam')

**>>>** a *# unique letters in a*

{'a', 'r', 'b', 'c', 'd'}

**>>>** a - b *# letters in a but not in b*

{'r', 'd', 'b'}

**>>>** a | b *# letters in a or b or both*

{'a', 'c', 'r', 'd', 'b', 'm', 'z', 'l'}

**>>>** a & b *# letters in both a and b*

{'a', 'c'}

**>>>** a ^ b *# letters in a or b but not both*

{'r', 'd', 'b', 'm', 'z', 'l'}

Similarly to [list comprehensions](https://docs.python.org/3/tutorial/datastructures.html#tut-listcomps), set comprehensions are also supported:

>>>

**>>>** a = {x **for** x **in** 'abracadabra' **if** x **not** **in** 'abc'}

**>>>** a

{'r', 'd'}

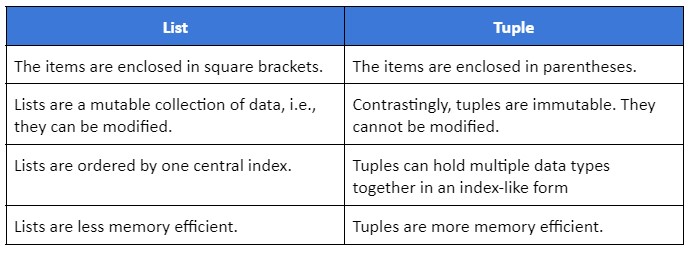
*In Python, you have heard that* ***lists, strings and tuples*** *are ordered collection of objects and* ***sets and dictionaries*** *are unordered collection of objects.*

* **Dictionaries**
* Another useful data type built into Python is the *dictionary* (see [Mapping Types — dict](https://docs.python.org/3/library/stdtypes.html#typesmapping)). Dictionaries are sometimes found in other languages as “associative memories” or “associative arrays”. Unlike sequences, which are indexed by a range of numbers, dictionaries are indexed by *keys*, which can be any immutable type; strings and numbers can always be keys. Tuples can be used as keys if they contain only strings, numbers, or tuples; if a tuple contains any mutable object either directly or indirectly, it cannot be used as a key. You can’t use lists as keys, since lists can be modified in place using index assignments, slice assignments, or methods like append() and extend().
* It is best to think of a dictionary as a set of *key: value* pairs, with the requirement that the keys are unique (within one dictionary). A pair of braces creates an empty dictionary: {}. Placing a comma-separated list of key:value pairs within the braces adds initial key:value pairs to the dictionary; this is also the way dictionaries are written on output.
* The main operations on a dictionary are storing a value with some key and extracting the value given the key. It is also possible to delete a key:value pair with del. If you store using a key that is already in use, the old value associated with that key is forgotten. It is an error to extract a value using a non-existent key.
* Performing list(d) on a dictionary returns a list of all the keys used in the dictionary, in insertion order (if you want it sorted, just use sorted(d) instead). To check whether a single key is in the dictionary, use the [in](https://docs.python.org/3/reference/expressions.html#in) keyword.
* Here is a small example using a dictionary:
* >>>
* **>>>** tel = {'jack': 4098, 'sape': 4139}
* **>>>** tel['guido'] = 4127
* **>>>** tel
* {'jack': 4098, 'sape': 4139, 'guido': 4127}
* **>>>** tel['jack']
* 4098
* **>>> del** tel['sape']
* **>>>** tel['irv'] = 4127
* **>>>** tel
* {'jack': 4098, 'guido': 4127, 'irv': 4127}
* **>>>** list(tel)
* ['jack', 'guido', 'irv']
* **>>>** sorted(tel)
* ['guido', 'irv', 'jack']
* **>>>** 'guido' **in** tel
* True
* **>>>** 'jack' **not** **in** tel
* False
* The [dict()](https://docs.python.org/3/library/stdtypes.html#dict) constructor builds dictionaries directly from sequences of key-value pairs:
* >>>
* **>>>** dict([('sape', 4139), ('guido', 4127), ('jack', 4098)])
* {'sape': 4139, 'guido': 4127, 'jack': 4098}
* In addition, dict comprehensions can be used to create dictionaries from arbitrary key and value expressions:
* >>>
* **>>>** {x: x\*\*2 **for** x **in** (2, 4, 6)}
* {2: 4, 4: 16, 6: 36}
* When the keys are simple strings, it is sometimes easier to specify pairs using keyword arguments:
* >>>
* **>>>** dict(sape=4139, guido=4127, jack=4098)
* {'sape': 4139, 'guido': 4127, 'jack': 4098}
* **Looping Techniques**
* When looping through dictionaries, the key and corresponding value can be retrieved at the same time using the items() method.
* >>>
* **>>>** knights = {'gallahad': 'the pure', 'robin': 'the brave'}
* **>>> for** k, v **in** knights.items():
* **...**  print(k, v)
* **...**
* gallahad the pure
* robin the brave
* When looping through a sequence, the position index and corresponding value can be retrieved at the same time using the [enumerate()](https://docs.python.org/3/library/functions.html#enumerate) function.
* **>>> for** i, v **in** enumerate(['tic', 'tac', 'toe']):
* **...**  print(i, v)
* **...**
* 0 tic
* 1 tac
* 2 toe
* To loop over two or more sequences at the same time, the entries can be paired with the [zip()](https://docs.python.org/3/library/functions.html#zip) function.
* **>>>** questions = ['name', 'quest', 'favorite color']
* **>>>** answers = ['lancelot', 'the holy grail', 'blue']
* **>>> for** q, a **in** zip(questions, answers):
* **...**  print('What is your **{0}**? It is **{1}**.'.format(q, a))
* **...**
* What is your name? It is lancelot.
* What is your quest? It is the holy grail.
* What is your favorite color? It is blue.
* To loop over a sequence in reverse, first specify the sequence in a forward direction and then call the [reversed()](https://docs.python.org/3/library/functions.html#reversed) function.
* >>>
* **>>> for** i **in** reversed(range(1, 10, 2)):
* **...**  print(i)
* **...**
* 9
* 7
* 5
* 3
* 1
* To loop over a sequence in sorted order, use the [sorted()](https://docs.python.org/3/library/functions.html#sorted) function which returns a new sorted list while leaving the source unaltered.
* >>>
* **>>>** basket = ['apple', 'orange', 'apple', 'pear', 'orange', 'banana']
* **>>> for** i **in** sorted(basket):
* **...**  print(i)
* **...**
* apple
* apple
* banana
* orange
* orange
* pear
* Using [set()](https://docs.python.org/3/library/stdtypes.html#set) on a sequence eliminates duplicate elements. The use of [sorted()](https://docs.python.org/3/library/functions.html#sorted) in combination with [set()](https://docs.python.org/3/library/stdtypes.html#set) over a sequence is an idiomatic way to loop over unique elements of the sequence in sorted order.
* >>>
* **>>>** basket = ['apple', 'orange', 'apple', 'pear', 'orange', 'banana']
* **>>> for** f **in** sorted(set(basket)):
* **...**  print(f)
* **...**
* apple
* banana
* orange
* pear
* It is sometimes tempting to change a list while you are looping over it; however, it is often simpler and safer to create a new list instead.
* >>>
* **>>> import** **math**
* **>>>** raw\_data = [56.2, float('NaN'), 51.7, 55.3, 52.5, float('NaN'), 47.8]
* **>>>** filtered\_data = []
* **>>> for** value **in** raw\_data:
* **...**  **if** **not** math.isnan(value):
* **...**  filtered\_data.append(value)
* **...**
* **>>>** filtered\_data
* [56.2, 51.7, 55.3, 52.5, 47.8]

### Enumerate differences between a list and a tuple in Python

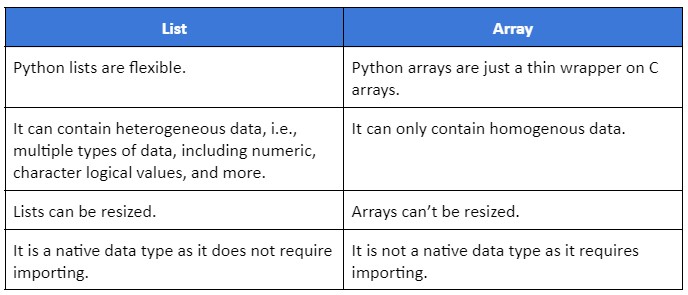
This is one of the basic Python data structures interview questions. Here’s how you can answer it:

The key differences between a list and a tuple are:



### How is a list different from an array?

The differences between lists and arrays are:



‍Learn the concept of [**Arrays in Data Structures**](https://www.interviewkickstart.com/learn/array-data-structure) here.

### Describe three advantages of NumPy arrays over Python lists

This is yet another popular Python data structure interview question. The three advantages of NumPy arrays over Python lists are:

* NumPy array is faster. The size of the NumPy arrays increases. It can become thirty times faster than Python lists.
* NumPy is more efficient and convenient. It comes with several vector and matrix operations for free, which helps avoid unnecessary work. Moreover, they can be efficiently implemented.
* Lastly, Python lists have certain limitations, like they don’t support element-wise addition, multiplication, and other vectorized operations. In addition, since lists contain heterogeneous objects, Python must store type information for every element. Contrastingly, arrays have homogeneous objects and thus escape these limitations.

### Why is Python a dynamically typed language?

Dynamic type checking means data types are checked during execution. Python is an interpreted language. It executes each statement line by line. So, type-checking is done during execution, making Python a dynamically typed language.

### What do you understand about inheritance in Python?

To answer this Python [**data structure interview question**](https://www.interviewkickstart.com/interview-questions/data-structure-interview-questions), you should know that inheritance is the property of one class to attain all the members (attributes and methods) of another class. Inheritance allows the reusability of code and makes it easier to create an application. It gives rise to two types of classes:

* **Superclass** is the class from which we are inheriting. It is also called the base class.
* **Derived Class** is the class that is inherited. It is also called the child class.

The various types of inheritance in Python are:

* **Single Inheritance** is when a derived class takes the members of a single superclass.
* **Multi-level inheritance**is when a derived class d1 is inherited from the base class- base1, and another derived class d2 is inherited from base2.
* **Hierarchical inheritance**allows the inheritance of a number of child classes from a single base class.
* **Multiple inheritances**are when a child class is inherited from more than one superclass.

### What do you understand about the join method in Python?

In Python, the join method is a string method. It takes elements of an iterable data structure (array, lists, and more) and connects them together using a string connector value.

### \*\*\* What are control flow statements in Python?

This is a common Python data structure interview question asked in tech interviews. This is how you can answer this particular question:

A program’s control flow refers to the order in which the program’s code executes. In Python, the control flow is regulated by conditional loops, statements, and function calls.

It has three main types of control structures:

* Sequential is the default mode
* Selection is used for decisions and branching
* Repetition helps in looping

### \*\*\*Explain memory management in Python.

Python private heap space manages memory, i.e., all objects and data structures of Python are located in a private heap. The python interpreter executes this heap, and no programmer has access to it. Python's memory manager allocates heap space for Python objects. Additionally, Python has an inbuilt garbage collector. It recycles all the unused memory.

### What are modules in Python? State a few benefits of modules.

This Python data structure interview question tests your basic understanding of the language. A Python module is a file containing a set of variables and functions that can be used in an application. The variables can be in the form of arrays, dictionaries, and objects.

Modules fall into two main categories:

* Built-in
* User-defined

Some key benefits of Python modules are:

* It allows structured code organization wherein code is logically grouped into a Python file. Thus, making development easier and less error-prone.
* Reusability of code as functionality in a single module can easily be reused. There is no need to recreate duplicate code.

Take a look at the [**Best Data Structures and Algorithms Course**](https://www.interviewkickstart.com/blog/best-data-structure-and-algorithms-course-for-faang-interview) to crack FAANG interviews.

### Explain slicing in Python.

Slicing is the mechanism to choose a range of items from sequence types such as lists, tuples, and strings. For example, slicing a list refers to selecting a specific portion or a subset of the list for some function, and the rest of the list remains unaffected. So, you remove a piece without altering the rest of the contents.

The syntax for slicing a list is: List\_name[start:stop:steps]

### What are decorators in Python?

In Python, decorators are essential functions. These add extension/ functionality to an existing function without altering the structure of the function itself. Instead, decorators take another function as their argument and return yet another function. This, too, is a common Python data structure interview question.

### State differences between xrange and range in Python.

Although xrange() and range() are similar functions in generating a sequence of integers. The major difference is that range returns a Python list of integers while xrange returns an xrange generator object. So xrange() does not generate a static list; instead, it creates the value on the go.

### What are generators in Python?

To answer this Python data structure interview question, understand that a generator is a function that returns an iterable collection of items. This happens one at a time. Thus, a generator is a special type of function that does not return a single value. Instead, they return an iterator object consisting of a sequence of values.

### What are Python keywords? Give examples of keywords.

Python has certain reserved words with special meanings called keywords. They define the type of variables. However, they cannot be used for variable or function names. There are 33 keywords in Python 3.0:

**Python Keywords**

* And
* Or
* Not
* As
* Def
* Lambda
* Pass
* Return
* If
* Elif
* Else
* For
* While
* Break
* True
* False
* Try
* With
* Assert
* Class
* Continue
* Except
* Finally
* From
* Global
* Yield
* Is
* Import
* In
* None
* Nonlocal
* Raise
* Del

### What are pickling and unpickling in Python?

This is how you can answer this Python data structure interview question:

The process of pickling in Python is when the object hierarchy is converted into a byte stream. Contrastingly, the process of unpickling is the inverse operation. Unpickling involves byte stream conversion back into an object hierarchy. Pickling allows you to arrange Python objects in a serial and allows de-serializing.

<https://www.educative.io/answers/what-are-pickling-and-unpickling-in-python>

### What is lambda in Python? State its uses.

In Python, lambda is an anonymous function. It can accept multiple arguments but has only a single expression. Lambda functions are used in situations needing an anonymous function for a short span of time. The uses of lambda functions are:

* They are used as small, single-line functions.
* They make code easier to read.

## Sorting algorithms

Top of Form

Bottom of Form

Top of Form

Bottom of Form

|  |  |
| --- | --- |
|  | ########################################## |
|  | # SORTING ALGORITHMS |
|  | # EXAMPLES, O(N), o(n) |
|  | # PROS, CONS |
|  | # Python code, Pseudo code |
|  | ########################################## |
|  | # https://www.freecodecamp.org/news/sorting-algorithms-explained/ |
|  | import sys |
|  | **############# Selection Sort #############** |
|  | # DESCRIPTION |
|  |  |
|  | # The selection sort algorithm sorts an array by repeatedly finding the minimum element (considering ascending order) |
|  | # from the unsorted part and putting it at the beginning. |
|  | # Follow the below steps to solve the problem: |
|  | # |
|  | # 1. Initialize minimum value(min\_idx) to location 0. |
|  | # 2. Traverse the array to find the minimum element in the array. |
|  | # 3. While traversing if any element smaller than min\_idx is found then swap both the values. |
|  | # 4. Then, increment min\_idx to point to the next element. |
|  | # 5. Repeat until the array is sorted. |
|  |  |
|  |  |
|  | # TIME COMPLEXITY |
|  |  |
|  | # Time Complexity: The time complexity of Selection Sort is O(N2) as there are two nested loops: |
|  | # 1. One loop to select an element of Array one by one = O(N) |
|  | # 2. Another loop to compare that element with every other Array element = O(N) |
|  | # Therefore, overall complexity = O(N) \* O(N) = O(N\*N) = O(N2) |
|  | # Auxiliary Space: O(1) as the only extra memory used is for temporary variables while swapping two values in Array. |
|  | # The selection sort never makes more than O(N) swaps and can be useful when memory write is a costly operation. |
|  |  |
|  | # STABILITY, IN PLACE |
|  |  |
|  | # Is Selection Sort Algorithm stable? |
|  | # Stability: The default implementation is not stable. However, it can be made stable. Please see stable selection sort for details. |
|  | # |
|  | # Is Selection Sort Algorithm in place? |
|  | # Yes, it does not require extra space. |
|  |  |
|  | **############# Bubble Sort ###################** |
|  | # DESCRIPTION |
|  | # Bubble Sort is the simplest sorting algorithm that works by repeatedly swapping the adjacent elements |
|  | # if they are in the wrong order. |
|  | # This algorithm is not suitable for large data sets as its average and worst-case time complexity is quite high. |
|  |  |
|  | # Steps: |
|  |  |
|  | # 1. Run a nested for loop to traverse the input array using two variables i and j, such that 0 ≤ i < n-1 and 0 ≤ j < n-i-1 |
|  | # 2. If arr[j] is greater than arr[j+1] then swap these adjacent elements, else move on |
|  |  |
|  | # PSEUDO CODE |
|  |  |
|  | # begin BubbleSortAlgorithm( Array ) |
|  | # |
|  | # For all the elements of the array |
|  | # |
|  | # if array[i] > array [i + 1] |
|  | # |
|  | # switch ( array[i] , array[i+!]) |
|  | # |
|  | # end if |
|  | # |
|  | # end for |
|  | # |
|  | # return array |
|  | # |
|  | #end BubbleSortAlgorithm |
|  |  |
|  | # TIME COMPLEXITY |
|  | # Time Complexity: O(N2) (nested loop) |
|  | # Auxiliary Space: O(1) |
|  | # |
|  | # Optimized Implementation of Bubble Sort: |
|  | # The above function always runs O(N2) time even if the array is sorted. |
|  | # It can be optimized by stopping the algorithm if the inner loop didn’t cause any swap |
|  |  |
|  |  |
|  | # Worst Case Analysis for Bubble Sort: |
|  | # The worst-case condition for bubble sort occurs when elements of the array are arranged in decreasing order. |
|  | # In the worst case, the total number of iterations or passes required to sort a given array is (n-1). |
|  | # where ‘n’ is a number of elements present in the array. |
|  | # |
|  | # At pass 1 : Number of comparisons = (n-1) |
|  | # Number of swaps = (n-1) |
|  | # |
|  | # At pass 2 : Number of comparisons = (n-2) |
|  | # Number of swaps = (n-2) |
|  | # |
|  | # At pass 3 : Number of comparisons = (n-3) |
|  | # Number of swaps = (n-3) |
|  | # . |
|  | # . |
|  | # . |
|  | # At pass n-1 : Number of comparisons = 1 |
|  | # Number of swaps = 1 |
|  | # |
|  | # Now, calculating total number of comparison required to sort the array |
|  | # = (n-1) + (n-2) + (n-3) + . . . 2 + 1 |
|  | # = (n-1)\*(n-1+1)/2 { by using sum of N natural Number formula } |
|  | # = n (n-1)/2 |
|  | # |
|  | # For the Worst case: |
|  | # Total number of swaps = Total number of comparison |
|  | # Total number of comparison (Worst case) = n(n-1)/2 |
|  | # Total number of swaps (Worst case) = n(n-1)/2 |
|  | # |
|  | # Worst and Average Case Time Complexity: O(N2). The worst case occurs when an array is reverse sorted. |
|  | # Best Case Time Complexity: O(N). The best case occurs when an array is already sorted. |
|  | # Auxiliary Space: O(1) |
|  |  |
|  | # Recursive Implementation Of Bubble Sort: |
|  | # The idea is to place the largest element in its position and keep doing the same for every other element. |
|  | # |
|  | # Follow the below steps to solve the problem: |
|  | # |
|  | # Place the largest element at its position, this operation makes sure that the first largest element will be placed at the end of the array. |
|  | # Recursively call for rest n – 1 elements with the same operation and place the next greater element at their position. |
|  | # The base condition for this recursion call would be, when the number of elements in the array becomes 0 or 1 then, simply return (as they are already sorted). |
|  | # Below is the implementation of the above approach: |
|  | # TBD |
|  |  |
|  | # STABILITY, IN PLACE |
|  |  |
|  | # What is the Boundary Case for Bubble sort? |
|  | # Bubble sort takes minimum time (Order of n) when elements are already sorted. |
|  | # Hence it is best to check if the array is already sorted or not beforehand, to avoid O(N2) time complexity. |
|  | # |
|  | # Does sorting happen in place in Bubble sort? |
|  | # Yes, Bubble sort performs swapping of adjacent pairs without the use of any major data structure. |
|  | # Hence Bubble sort algorithm is an in-place algorithm. |
|  |  |
|  | # Is the Bubble sort algorithm stable? |
|  | # Yes, the bubble sort algorithm is stable. |
|  |  |
|  | # Where is the Bubble sort algorithm used? |
|  | # Due to its simplicity, bubble sort is often used to introduce the concept of a sorting algorithm. |
|  | # In computer graphics, it is popular for its capability to detect a tiny error |
|  | # (like a swap of just two elements) in almost-sorted arrays and fix it with just linear |
|  | # complexity (2n). |
|  |  |
|  | # Example: It is used in a polygon filling algorithm, where bounding lines are sorted by their x coordinate |
|  | # at a specific scan line (a line parallel to the x-axis), and with incrementing y their order changes (two elements are swapped) |
|  | # only at intersections of two lines (Source: Wikipedia) |
|  |  |
|  | **############# Insertion Sort ###################** |
|  |  |
|  | # DESCRIPTION |
|  | # Insertion sort is a simple sorting algorithm that works similar to the way you sort playing cards in your hands. |
|  | # The array is virtually split into a sorted and an unsorted part. |
|  | # Values from the unsorted part are picked and placed at the correct position in the sorted part. |
|  | # |
|  | # Characteristics of Insertion Sort: |
|  | # 1. This algorithm is one of the simplest algorithm with simple implementation |
|  | # 2. Basically, Insertion sort is efficient for small data values |
|  | # 3. Insertion sort is adaptive in nature, i.e. it is appropriate for data sets which are already partially sorted. |
|  |  |
|  | # Insertion Sort Algorithm |
|  | # To sort an array of size N in ascending order: |
|  | # |
|  | # 1. Iterate from arr[1] to arr[N] over the array. |
|  | # 2. Compare the current element (key) to its predecessor. |
|  | # 3. If the key element is smaller than its predecessor, compare it to the elements before. |
|  | # 4. Move the greater elements one position up to make space for the swapped element. |
|  |  |
|  |  |
|  | # PSEUDO CODE |
|  | # https://www.ecb.torontomu.ca/~courses/coe428/sorting/insertionsort.html#:~:text=The%20pseudocode%20for%20insertion%20sort,by%20length%5BA%5D.) |
|  |  |
|  | # INSERTION-SORT (A) |
|  | # for j <- 2 to length[A] |
|  | #2 do key <- A[j] |
|  | #3 Insert A[j] into the sorted sequence A[1 . . j - 1]. |
|  | #4 i <- j - 1 |
|  | #5 while i > 0 and A[i] > key |
|  | #6 do A[i + 1] <- A[i] |
|  | #7 i <- i - 1 |
|  | #8 A[i + 1] <- key |
|  |  |
|  | # TIME COMPLEXITY |
|  |  |
|  | # Time Complexity: O(N^2) (nested loop) |
|  | # Auxiliary Space: O(1) |
|  |  |
|  | # STABILITY, IN PLACE |
|  |  |
|  | # 1. What are the Boundary Cases of the Insertion Sort algorithm? |
|  | # Insertion sort takes maximum time to sort if elements are sorted in reverse order. |
|  | # And it takes minimum time (Order of n) when elements are already sorted. |
|  | # |
|  | # 2. What are the Algorithmic Paradigm of Insertion Sort algorithm? |
|  | # Insertion Sort algorithm follows incremental approach. |
|  | # |
|  | # 3. Is Insertion Sort an in-place sorting algorithm? |
|  | # Yes, insertion sort is an in-place sorting algorithm. |
|  | # |
|  | # 4. Is Insertion Sort a stable algorithm? |
|  | # Yes, insertion sort is a stable sorting algorithm. |
|  | # |
|  | # 5. When is the Insertion Sort algorithm used? |
|  | # Insertion sort is used when number of elements is small. |
|  | # It can also be useful when input array is almost sorted, only few elements are misplaced in complete big array. |
|  | # |
|  | #6. What is Binary Insertion Sort? |
|  | # We can use binary search to reduce the number of comparisons in normal insertion sort. |
|  | # Binary Insertion Sort uses binary search to find the proper location to insert the selected item at each iteration. |
|  | # In normal insertion, sorting takes O(i) (at ith iteration) in worst case. We can reduce it to O(logi) by using binary search. |
|  | # The algorithm, as a whole, still has a running worst case running time of O(n^2) |
|  | # because of the series of swaps required for each insertion. |
|  | # Implementation of sort for Linked List |
|  | # https://www.geeksforgeeks.org/insertion-sort-for-singly-linked-list/ |
|  |  |
|  | # Create an empty sorted (or result) list |
|  | # Traverse the given list, do following for every node. |
|  | # Insert current node in sorted way in sorted or result list. |
|  | # Change head of given linked list to head of sorted (or result) list. |
|  |  |
|  | **############# Merge Sort #######################** |
|  | # DESCRIPTION |
|  | # The Merge Sort algorithm is a sorting algorithm that is based on the Divide and Conquer paradigm. |
|  | # In this algorithm, the array is initially divided into two equal halves and then they are combined in a sorted manner. |
|  | # |
|  | # Merge Sort Working Process: |
|  | # Think of it as a recursive algorithm continuously splits the array in half until it cannot be further divided. |
|  | # This means that if the array becomes empty or has only one element left, the dividing will stop, |
|  | # i.e. it is the base case to stop the recursion. |
|  | # If the array has multiple elements, split the array into halves and recursively invoke the merge sort on each of the halves. |
|  | # Finally, when both halves are sorted, the merge operation is applied. |
|  | # Merge operation is the process of taking two smaller sorted arrays and combining them to eventually make a larger one. |
|  |  |
|  |  |
|  | # PSEUDO CODE |
|  | # |
|  | #step 1: start |
|  | # |
|  | #step 2: declare array and left, right, mid variable |
|  | # |
|  | #step 3: perform merge function. |
|  | # if left > right |
|  | # return |
|  | # mid= (left+right)/2 |
|  | # mergesort(array, left, mid) |
|  | # mergesort(array, mid+1, right) |
|  | # merge(array, left, mid, right) |
|  | # |
|  | #step 4: Stop |
|  |  |
|  | #Follow the steps below the solve the problem: |
|  | # |
|  | #MergeSort(arr[], l, r) |
|  | #If r > l |
|  | # |
|  | #Find the middle point to divide the array into two halves: |
|  | #middle m = l + (r – l)/2 |
|  | #Call mergeSort for first half: |
|  | #Call mergeSort(arr, l, m) |
|  | #Call mergeSort for second half: |
|  | #Call mergeSort(arr, m + 1, r) |
|  | #Merge the two halves sorted in steps 2 and 3: |
|  | #Call merge(arr, l, m, r) |
|  |  |
|  | # TIME COMPLEXITY |
|  |  |
|  | #Time Complexity: O(N log(N)), |
|  | #Sorting arrays on different machines. |
|  | #Merge Sort is a recursive algorithm and time complexity can be expressed as following recurrence relation. |
|  | # |
|  | #T(n) = 2T(n/2) + θ(n) |
|  | # |
|  | #The above recurrence can be solved either using the Recurrence Tree method or the Master method. |
|  | #It falls in case II of the Master Method and the solution of the recurrence is θ(Nlog(N)). |
|  | #The time complexity of Merge Sort isθ(Nlog(N)) in all 3 cases (worst, average, and best) |
|  | #as merge sort always divides the array into two halves and takes linear time to merge two halves. |
|  | # |
|  | #Auxiliary Space: O(n), In merge sort all elements are copied into an auxiliary array. |
|  | #So N auxiliary space is required for merge sort. |
|  |  |
|  | # STABILITY, IN-PLACE |
|  |  |
|  | #Is Merge sort In Place? |
|  | #No, In merge sort the merging step requires extra space to store the elements. |
|  | # |
|  | #Is Merge sort Stable? |
|  | #Yes, merge sort is stable. |
|  | # |
|  | #How can we make Merge sort more efficient? |
|  | #Merge sort can be made more efficient by replacing recursive calls with Insertion sort for smaller array sizes, |
|  | #where the size of the remaining array is less or equal to 43 as the number of operations required to sort an array |
|  | #of max size 43 will be less in Insertion sort as compared to the number of operations required in Merge sort. |
|  | # |
|  | #Analysis of Merge Sort: |
|  | #A merge sort consists of several passes over the input. The first pass merges segments of size 1, |
|  | #the second merges segments of size 2, and the i\_{th} pass merges segments of size 2i-1. |
|  | #Thus, the total number of passes is [log2n]. |
|  | #As merge showed, we can merge two sorted segments in linear time, which means that each pass takes O(n) time. |
|  | #Since there are [log2n] passes, the total computing time is O(nlogn). |
|  |  |
|  | **######################### Quick Sort #########################################** |
|  |  |
|  | # DESCRIPTION |
|  | #Like Merge Sort, QuickSort is a Divide and Conquer algorithm. |
|  | #It picks an element as a pivot and partitions the given array around the picked pivot. |
|  | #There are many different versions of quickSort that pick pivot in different ways. |
|  | # |
|  | #Always pick the first element as a pivot. |
|  | #Always pick the last element as a pivot (implemented below) |
|  | #Pick a random element as a pivot. |
|  | #Pick median as the pivot. |
|  | #The key process in quickSort is a partition(). |
|  | #The target of partitions is, given an array and an element x of an array as the pivot, |
|  | #put x at its correct position in a sorted array and put all smaller elements (smaller than x) before x, |
|  | #and put all greater elements (greater than x) after x. All this should be done in linear time. |
|  |  |
|  |  |
|  | # COMPLEXITY |
|  |  |
|  | #Time taken by QuickSort, in general, can be written as follows. |
|  | # |
|  | # T(n) = T(k) + T(n-k-1) + \theta (n) |
|  | # |
|  | #The first two terms are for two recursive calls, the last term is for the partition process. |
|  | # k is the number of elements that are smaller than the pivot. |
|  | #The time taken by QuickSort depends upon the input array and partition strategy. Following are three cases. |
|  | # |
|  | #Worst Case: |
|  | #The worst case occurs when the partition process always picks the greatest or smallest element as the pivot. |
|  | #If we consider the above partition strategy where the last element is always picked as a pivot, |
|  | #the worst case would occur when the array is already sorted in increasing or decreasing order. |
|  | #Following is recurrence for the worst case. |
|  | # |
|  | # T(n) = T(0) + T(n-1) + \theta(n) |
|  | #which is equivalent to T(n) = T(n-1) + \theta(n) |
|  | # |
|  | #The solution to the above recurrence is (n2). |
|  | # |
|  |  |
|  | # STABILITY, IN-PLACE |
|  | # |
|  | #Is QuickSort stable? |
|  | #The default implementation is not stable. |
|  | #However any sorting algorithm can be made stable by considering indexes as comparison parameter. |
|  | # |
|  | #Is QuickSort In-place? |
|  | #As per the broad definition of in-place algorithm it qualifies as an in-place sorting algorithm |
|  | #as it uses extra space only for storing recursive function calls but not for manipulating the input. |
|  |  |
|  | **##################### Heap Sort ###########################################################################** |
|  |  |
|  | # DESCRIPTION |
|  |  |
|  | #Heap sort is a comparison-based sorting technique based on Binary Heap data structure. |
|  | # It is similar to the selection sort where we first find the minimum element and place the minimum element at the beginning. |
|  | # Repeat the same process for the remaining elements. |
|  | # |
|  | #Heap sort is an in-place algorithm. |
|  | #Its typical implementation is not stable, but can be made stable (See this) |
|  | #Typically 2-3 times slower than well-implemented QuickSort. The reason for slowness is a lack of locality of reference. |
|  | #Advantages of heapsort: |
|  | #Efficiency – The time required to perform Heap sort increases logarithmically |
|  | #while other algorithms may grow exponentially slower as the number of items to sort increases. |
|  | #This sorting algorithm is very efficient. |
|  | #Memory Usage – Memory usage is minimal because apart from what is necessary to hold the initial list of items to be sorted, |
|  | #it needs no additional memory space to work |
|  | #Simplicity – It is simpler to understand than other equally efficient sorting algorithms because |
|  | #it does not use advanced computer science concepts such as recursion |
|  | #Applications of HeapSort: |
|  | #Heapsort is mainly used in hybrid algorithms like the IntroSort. |
|  | #Sort a nearly sorted (or K sorted) array |
|  | #k largest(or smallest) elements in an array |
|  | #The heap sort algorithm has limited uses because Quicksort and Mergesort are better in practice. |
|  | #Nevertheless, the Heap data structure itself is enormously used. |
|  |  |
|  | #What is meant by Heapify? |
|  | #Heapify is the process of creating a heap data structure from a binary tree represented using an array. |
|  | # It is used to create Min-Heap or Max-heap. Start from the first index of the non-leaf node whose index is given by n/2 – 1. |
|  | # Heapify uses recursion |
|  |  |
|  |  |
|  | # TIME COMPLEXITY |
|  |  |
|  | #Time Complexity: O(N log N) |
|  | #Auxiliary Space: O(1) |
|  |  |
|  | # STABILITY |
|  |  |
|  | #What are the two phases of Heap Sort? |
|  | #The heap sort algorithm consists of two phases. In the first phase the array is converted into a max heap. |
|  | # And in the second phase the highest element is removed (i.e., the one at the tree root) and |
|  | # the remaining elements are used to create a new max heap. |
|  | # |
|  | #Why Heap Sort is not stable? |
|  | #Heap sort algorithm is not a stable algorithm. This algorithm is not stable because the operations |
|  | # that are performed in a heap can change the relative ordering of the equivalent keys. |
|  | # |
|  | #Is Heap Sort an example of “Divide and Conquer” algorithm? |
|  | #Heap sort is NOT at all a Divide and Conquer algorithm. It uses a heap data structure to efficiently sort its element |
|  | #and not a “divide and conquer approach” to sort the elements. |
|  | # |
|  | #Which sorting algorithm is better – Heap sort or Merge Sort? |
|  | #The answer lies in the comparison of their time complexity and space requirement. |
|  | #The Merge sort is slightly faster than the Heap sort. But on the other hand merge sort takes extra memory. Depending on the requirement, one should choose which one to use. |
|  | # |
|  | #Why Heap sort better than Selection sort? |
|  | #Heap sort is similar to selection sort, but with a better way to get the maximum element. |
|  | # It takes advantage of the heap data structure to get the maximum element in constant time. |
|  |  |
|  | **########################### Counting Sort ###########################** |
|  |  |
|  | # DESCRIPTION |
|  |  |
|  |  |
|  |  |
|  |  |
|  | **####################### Radix Sort ########################################** |
|  |  |
|  | # DESCRIPTION |
|  |  |
|  |  |
|  | **####################### Bucket Sort ########################################** |
|  |  |
|  | # DESCRIPTION |
|  |  |

## Other algorithms, algorithm efficiency

**Strings Manipulation**

**1. Reverse Integer**

|  |  |  |
| --- | --- | --- |
| # Given an integer, return the integer with reversed digits. | | |
|  | # Note: The integer could be either positive or negative. |
|  |  |
|  | def solution(x): |
|  | string = str(x) |
|  |  |
|  | if string[0] == '-': |
|  | return int('-'+string[:0:-1]) |
|  | else: |
|  | return int(string[::-1]) |
|  |  |
|  | print(solution(-231)) |
|  | print(solution(345)) |

A warm-up algorithm, that will help you practicing your slicing skills. In effect the only tricky bit is to make sure you are taking into account the case when the integer is negative. I have seen this problem presented in many different ways but it usually is the starting point for more complex requests.

**2. Average Words Length**

|  |  |
| --- | --- |
| # For a given sentence, return the average word length. | |
|  | # Note: Remember to remove punctuation first. |
|  |  |
|  | sentence1 = "Hi all, my name is Tom...I am originally from Australia." |
|  | sentence2 = "I need to work very hard to learn more about algorithms in Python!" |
|  |  |
|  | def solution(sentence): |
|  | for p in "!?',;.": |
|  | sentence = sentence.replace(p, '') |
|  | words = sentence.split() |
|  | return round(sum(len(word) for word in words)/len(words),2) |
|  |  |
|  | print(solution(sentence1)) |
|  | print(solution(sentence2)) |

Algorithms that require you to apply some simple calculations using strings are very common, therefore it is important to get familiar with methods like .**replace**() and .**split**() that in this case helped me removing the unwanted characters and create a list of words, the length of which can be easily measured and summed.

**3. Add Strings**

|  |  |  |
| --- | --- | --- |
| # Given two non-negative integers num1 and num2 represented as string, return the sum of num1 and num2. | | |
|  | # You must not use any built-in BigInteger library or convert the inputs to integer directly. |
|  |  |
|  | #Notes: |
|  | #Both num1 and num2 contains only digits 0-9. |
|  | #Both num1 and num2 does not contain any leading zero. |
|  |  |
|  | num1 = '364' |
|  | num2 = '1836' |
|  |  |
|  | # Approach 1: |
|  | def solution(num1,num2): |
|  | eval(num1) + eval(num2) |
|  | return str(eval(num1) + eval(num2)) |
|  |  |
|  | print(solution(num1,num2)) |
|  |  |
|  | #Approach2 |
|  | #Given a string of length one, the ord() function returns an integer representing the Unicode code point of the character |
|  | #when the argument is a unicode object, or the value of the byte when the argument is an 8-bit string. |
|  |  |
|  | def solution(num1, num2): |
|  | n1, n2 = 0, 0 |
|  | m1, m2 = 10\*\*(len(num1)-1), 10\*\*(len(num2)-1) |
|  |  |
|  | for i in num1: |
|  | n1 += (ord(i) - ord("0")) \* m1 |
|  | m1 = m1//10 |
|  |  |
|  | for i in num2: |
|  | n2 += (ord(i) - ord("0")) \* m2 |
|  | m2 = m2//10 |
|  |  |
|  | return str(n1 + n2) |
|  | print(solution(num1, num2)) |

I find both approaches equally sharp: the first one for its brevity and the intuition of using the eval( )method to dynamically evaluate string-based inputs and the second one for the smart use of the ord( ) function to re-build the two strings as actual numbers trough the Unicode code points of their characters. If I really had to chose in between the two, I would probably go for the second approach as it looks more complex at first but it often comes handy in solving “Medium” and “Hard” algorithms that require more advanced string manipulation and calculations.

**4. First Unique Character**

|  |  |
| --- | --- |
| # Given a string, find the first non-repeating character in it and return its index. | |
|  | # If it doesn't exist, return -1. # Note: all the input strings are already lowercase. |
|  |  |
|  | #Approach 1 |
|  | def solution(s): |
|  | frequency = {} |
|  | for i in s: |
|  | if i not in frequency: |
|  | frequency[i] = 1 |
|  | else: |
|  | frequency[i] +=1 |
|  | for i in range(len(s)): |
|  | if frequency[s[i]] == 1: |
|  | return i |
|  | return -1 |
|  |  |
|  | print(solution('alphabet')) |
|  | print(solution('barbados')) |
|  | print(solution('crunchy')) |
|  |  |
|  | print('###') |
|  |  |
|  | #Approach 2 |
|  | import collections |
|  |  |
|  | def solution(s): |
|  | # build hash map : character and how often it appears |
|  | count = collections.Counter(s) # <-- gives back a dictionary with words occurrence count |
|  | #Counter({'l': 1, 'e': 3, 't': 1, 'c': 1, 'o': 1, 'd': 1}) |
|  | # find the index |
|  | for idx, ch in enumerate(s): |
|  | if count[ch] == 1: |
|  | return idx |
|  | return -1 |
|  |  |
|  | print(solution('alphabet')) |
|  | print(solution('barbados')) |
|  | print(solution('crunchy')) |

Also in this case, two potential solutions are provided and I guess that, if you are pretty new to algorithms, the first approach looks a bit more familiar as it builds as simple counter starting from an empty dictionary.

However understanding the second approach will help you much more in the longer term and this is because in this algorithm I simply used collection.Counter(s)instead of building a chars counter myself and replaced range(len(s)) with enumerate(s), a function that can help you identify the index more elegantly.

**5. Valid Palindrome**

|  |  |  |
| --- | --- | --- |
| # Given a non-empty string s, you may delete at most one character. Judge whether you can make it a palindrome. | | |
|  | # The string will only contain lowercase characters a-z. |
|  |  |
|  | s = 'radkar' |
|  | def solution(s): |
|  | for i in range(len(s)): |
|  | t = s[:i] + s[i+1:] |
|  | if t == t[::-1]: return True |
|  |  |
|  | return s == s[::-1] |
|  |  |
|  | solution(s) |

The “Valid Palindrome” problem is a real classic and you will probably find it repeatedly under many different flavors. In this case, the task is to check weather by removing at most one character, the string matches with its reversed counterpart. When s = ‘radkar’ the function returns Trueas by excluding the ‘k’ we obtain the word ‘radar’ that is a palindrome.

**Arrays**

**6. Monotonic Array**

|  |  |  |
| --- | --- | --- |
| # Given an array of integers, determine whether the array is monotonic or not. | | |
|  | A = [6, 5, 4, 4] |
|  | B = [1,1,1,3,3,4,3,2,4,2] |
|  | C = [1,1,2,3,7] |
|  |  |
|  | def solution(nums): |
|  | return (all(nums[i] <= nums[i + 1] for i in range(len(nums) - 1)) or |
|  | all(nums[i] >= nums[i + 1] for i in range(len(nums) - 1))) |
|  |  |
|  | print(solution(A)) |
|  | print(solution(B)) |
|  | print(solution(C)) |

This is another very frequently asked problem and the solution provided above is pretty elegant as it can be written as a one-liner. An array is monotonic if and only if it is monotone increasing, or monotone decreasing and in order to assess it, the algorithm above takes advantage of the all() function that returns Trueif all items in an iterable are true, otherwise it returns False. If the iterable object is empty, the all() function also returns True.

**7. Move Zeroes**

|  |  |  |
| --- | --- | --- |
| #Given an array nums, write a function to move all zeroes to the end of it while maintaining the relative order of | | |
|  | #the non-zero elements. |
|  |  |
|  | array1 = [0,1,0,3,12] |
|  | array2 = [1,7,0,0,8,0,10,12,0,4] |
|  |  |
|  | def solution(nums): |
|  | for i in nums: |
|  | if 0 in nums: |
|  | nums.remove(0) |
|  | nums.append(0) |
|  | return nums |
|  |  |
|  | solution(array1) |
|  | solution(array2) |

When you work with arrays, the .remove() and .append() methods are precious allies. In this problem I have used them to first remove each zero that belongs to the original array and then append it at the end to the same array.

**8. Fill The Blanks**

|  |  |  |
| --- | --- | --- |
| # Given an array containing None values fill in the None values with most recent | | |
|  | # non None value in the array |
|  | array1 = [1,None,2,3,None,None,5,None] |
|  |  |
|  | def solution(array): |
|  | valid = 0 |
|  | res = [] |
|  | for i in nums: |
|  | if i is not None: |
|  | res.append(i) |
|  | valid = i |
|  | else: |
|  | res.append(valid) |
|  | return res |
|  |  |
|  | solution(array1) |

I was asked to solve this problem a couple of times in real interviews, both times the solution had to include edge cases (that I omitted here for simplicity). On paper, this an easy algorithm to build but you need to have clear in mind what you want to achieve with the for loop and if statement and be comfortable working with None values.

**9. Matched & Mismatched Words**

|  |  |
| --- | --- |
| #Given two sentences, return an array that has the words that appear in one sentence and not | |
|  | #the other and an array with the words in common. |
|  |  |
|  | sentence1 = 'We are really pleased to meet you in our city' |
|  | sentence2 = 'The city was hit by a really heavy storm' |
|  |  |
|  | def solution(sentence1, sentence2): |
|  | set1 = set(sentence1.split()) |
|  | set2 = set(sentence2.split()) |
|  |  |
|  | return sorted(list(set1^set2)), sorted(list(set1&set2)) # ^ A.symmetric\_difference(B), & A.intersection(B) |
|  |  |
|  | print(solution(sentence1, sentence2)) |

The problem is fairly intuitive but the algorithm takes advantage of a few very common set operations like set() , intersection() or &and symmetric\_difference()or ^that are extremely useful to make your solution more elegant. If it is the first time you encounter them, make sure to check this article:

**10. Prime Numbers Array**

|  |  |
| --- | --- |
| # Given k numbers which are less than n, return the set of prime number among them | |
|  | # Note: The task is to write a program to print all Prime numbers in an Interval. |
|  | # Definition: A prime number is a natural number greater than 1 that has no positive divisors other than 1 and itself. |
|  |  |
|  | n = 35 |
|  | def solution(n): |
|  | prime\_nums = [] |
|  | for num in range(n): |
|  | if num > 1: # all prime numbers are greater than 1 |
|  | for i in range(2, num): |
|  | if (num % i) == 0: # if the modulus == 0 is means that the number can be divided by a number preceding it |
|  | break |
|  | else: |
|  | prime\_nums.append(num) |
|  | return prime\_nums |
|  | solution(n) |

I wanted to close this section with another classic problem. A solution can be found pretty easily looping trough range(n) if you are familiar with both the prime numbers definition and the modulus operation.

**1. What are the basic operations that can be performed with a list?**

The basic operations that can be performed with a list are:

– Append: Add an element to the end of the list

– Insert: Add an element at a specific index

– Remove: Remove an element from the list

– Index: Get the index of an element

– Sort: Sort the list in ascending or descending order

– Reverse: Reverse the order of the list

**2. How do you create an empty list in Python?**

There are a few ways to create an empty list in Python. One way is to use the list() function without any arguments. Another way is to use the [] brackets to create an empty list.

**3. Is it possible to reverse a list using slicing only? If yes, then how?**

Yes, it is possible to reverse a list using slicing only. To do so, you would need to use a negative step size when slicing the list. For example, if you have a list called my\_list, you could reverse it using my\_list[::-1].

**4. Can you explain what “list comprehension” means?**

List comprehension is a feature of Python that allows you to create lists in a more concise way than the standard method of using a for loop. With list comprehension, you can define the list you want to create and the conditions for how that list should be populated all in one line of code.

**5. What is the difference between sorting and reversing a list?**

Sorting a list means putting the list in order from smallest to largest (or vice versa). Reversing a list means taking the list and reversing the order of the elements. So, if the list was [1,2,3,4], reversing it would give you [4,3,2,1].

**6. Why should we use dictionaries instead of lists? When would one choose a dictionary over a list?**

Dictionaries are more efficient than lists when you need to store data that you will need to look up quickly. Dictionaries are also more flexible than lists, because you can use any immutable data type as a key, not just integers.

**7. What’s the best way to check for duplicates in a list?**

There are a few ways to check for duplicates in a list, but the most efficient way is to use a set. A set is a data structure that only allows unique values, so by converting the list into a set, you can easily check if there are any duplicates.

**8. How does a binary search tree work?**

A binary search tree is a data structure that allows for efficient search and retrieval of data. Each node in the tree stores a value, and the left and right child nodes of each node contain values that are less than or greater than the node’s value, respectively. This allows for quick and efficient search of the tree, as the search can be narrowed down to only those nodes that contain values that are relevant to the search.

**9. What are some of the important properties of a binary search tree?**

A binary search tree is a data structure that allows for efficient search, insertion, and deletion of data. Each node in the tree has at most two child nodes, and the left child node is always less than the right child node. This structure allows for quick and efficient search of data.

**10. What is the maximum number of nodes allowed in a full binary tree?**

A full binary tree is a tree in which each node has either 0 or 2 children. The maximum number of nodes in a full binary tree is 2^n – 1, where n is the number of levels in the tree.

**11. What is a circular linked list?**

A circular linked list is a type of linked list where the last node in the list points back to the first node, creating a loop. This can be useful for certain applications where you need to be able to loop through the data structure.

**12. Can you give me a few examples of situations where a circular linked list might make more sense than a regular linked list?**

A circular linked list can be used when you need to keep track of a list of items where the order is not important, but you need to be able to quickly move from one item to the next. For example, if you were maintaining a list of active users on a website, a circular linked list would allow you to quickly move from one user to the next without having to keep track of the order.

**13. What types of traversal methods are supported by trees in general?**

The three main types of traversal methods supported by trees are pre-order, in-order, and post-order.

**14. What is recursion? Give a few real-world examples of when it’s useful.**

Recursion is a programming technique that allows a function to call itself. This can be useful for tasks that can be divided into smaller sub-tasks, such as parsing a document or traversing a data structure. Some real-world examples of when recursion is useful include:

– Searching through a large document or data structure for a specific value

– Parsing a complicated data structure

– Generating a complex report or document

**15. What is memoization? Give a few real-world examples of when it’s useful.**

**Memoization** is a technique for improving the performance of a program by caching the results of expensive function calls and reusing the cached results when the same inputs occur again. Memoization can be used to speed up algorithms that perform the same calculation multiple times, such as in dynamic programming or recursion. Real-world examples of memoization include caching the results of database queries, web service calls, or heavy computations.

**16. What is the time complexity of searching through a sorted array using a binary search algorithm?**

The time complexity of a binary search algorithm is O(log n), where n is the number of elements in the array. This is because the algorithm only needs to look at the middle element of the array in order to determine whether to search the left or right side of the array.

**17. What is the time complexity of performing insertion on a HashTable data structure?**

The time complexity of insertion into a HashTable is O(1). This is because insertion simply involves adding a new key-value pair to the HashTable, which can be done in constant time.

**18. What is the time complexity of finding the largest element in a binary heap?**

The time complexity of finding the largest element in a binary heap is O(log n). This is because, in order to find the largest element, you need to compare the root element to its children. However, because the binary heap is a complete binary tree, the number of comparisons is limited to the height of the tree, which is log n.

**19. What is the time complexity of adding or removing elements from a Queue?**

The time complexity of adding or removing elements from a Queue is O(1).

**20. What is the time complexity of inserting an element into a Linked List?**

The time complexity of inserting an element into a Linked List is O(1). This is because the operation can be performed in a single step, regardless of the size of the Linked List.

## loc/iloc

The main distinction between the two methods is:

* loc gets rows (and/or columns) with particular **labels**.
* iloc gets rows (and/or columns) at integer **locations**.

To demonstrate, consider a series s of characters with a non-monotonic integer index:

>>> s = pd.Series(list("abcdef"), index=[49, 48, 47, 0, 1, 2])

49 a

48 b

47 c

0 d

1 e

2 f

>>> s.loc[0] # value at index label 0

'd'

>>> s.iloc[0] # value at index location 0

'a'

>>> s.loc[0:1] # rows at index labels between 0 and 1 (inclusive)

0 d

1 e

>>> s.iloc[0:1] # rows at index location between 0 and 1 (exclusive)

49 a

Here are some of the differences/similarities between s.loc and s.iloc when passed various objects:

| **<object>** | **description** | **s.loc[<object>]** | **s.iloc[<object>]** |
| --- | --- | --- | --- |
| 0 | single item | Value at index label 0 (the string 'd') | Value at index location 0 (the string 'a') |
| 0:1 | slice | **Two** rows (labels 0 and 1) | **One** row (first row at location 0) |
| 1:47 | slice with out-of-bounds end | **Zero** rows (empty Series) | **Five** rows (location 1 onwards) |
| 1:47:-1 | slice with negative step | **three** rows (labels 1 back to 47) | **Zero** rows (empty Series) |
| [2, 0] | integer list | **Two** rows with given labels | **Two** rows with given locations |
| s > 'e' | Bool series (indicating which values have the property) | **One** row (containing 'f') | NotImplementedError |
| (s>'e').values | Bool array | **One** row (containing 'f') | Same as loc |
| 999 | int object not in index | KeyError | IndexError (out of bounds) |
| -1 | int object not in index | KeyError | Returns last value in s |
| lambda x: x.index[3] | callable applied to series (here returning 3rd item in index) | s.loc[s.index[3]] | s.iloc[s.index[3]] |

loc's label-querying capabilities extend well-beyond integer indexes and it's worth highlighting a couple of additional examples.

## What is a set function?

Set, a term in mathematics for a sequence consisting of distinct language is also extended in its language by Python and can easily be made using set().

set() method is used to convert any of the iterable to sequence of iterable elements with distinct elements, commonly called Set.

Syntax : set(iterable)

Parameters : Any iterable sequence like list, tuple or dictionary.

Returns : An empty set if no element is passed. Non-repeating element iterable modified as passed as argument.

## Can you add a list to the dictionary?

The same way you would add any other object.

myDict = {}

myDict[“myList”] = []

That would add an empty list, whose key is “myList”, to the dictionary “myDict”

You could also use the dict “update” method, as follows:

myDict.update( myList: [] )

or this way

myDict.update( {‘myList’: []} )

Although the second method is more appropriate to add several key/value pairs or for merging two dictionaries.

## What does x[-1] do?

Get the last item of the list

## What does x[::-1] do?

It reverts the list.

## OOP – classes/inheritance

**What are Classes and Objects?**

Python, like every other object-oriented language, allows you to define classes to create objects. In-built Python classes are the most common data types in Python, such as strings, lists, dictionaries, and so on.

**A class** is a **collection of** **instance variables and related methods** that define a particular object type. You can think of a class as an object's blueprint or template. Attributes are the names given to the variables that make up a class.

**A class instance** with a **defined set of properties** is called an **object**. As a result, the same class can be used to construct as many objects as needed.

Let’s define a class named Book for a bookseller’s sales software.

*class Book:*

*def \_\_init\_\_(self, title, quantity, author, price):*

*self.title = title*

*self.quantity = quantity*

*self.author = author*

*self.price = price*

The *\_\_init\_\_* special method, also known as a **Constructor**, is used to initialize the Book class with attributes such as title, quantity, author, and price.

In Python, **built-in classes** are named in lower case, but user-defined classes are named in Camel or Snake case, with the first letter capitalized.

This class can be instantiated to any number of objects. Three books are instantiated in the following example code:

*book1 = Book('Book 1', 12, 'Author 1', 120)*

*book2 = Book('Book 2', 18, 'Author 2', 220)*

*book3 = Book('Book 3', 28, 'Author 3', 320)*

*book1, book2 and book3* are distinct objects of the class Book. The term *self* in the attributes refers to the corresponding instances (objects).

*print(book1)*

*print(book2)*

*print(book3)*

*Output:*

*<\_\_main\_\_.Book object at 0x00000156EE59A9D0>*

*<\_\_main\_\_.Book object at 0x00000156EE59A8B0>*

*<\_\_main\_\_.Book object at 0x00000156EE59ADF0>*

**The class and memory location** of the objects are printed when they are printed. We can't expect them to provide specific information on the qualities, such as the title, author name, and so on. But we can use a specific method called *\_\_repr\_\_* to do this.

In Python, a special method is a defined function that starts and ends with two underscores and is invoked automatically when certain conditions are met.

*class Book:*

*def \_\_init\_\_(self, title, quantity, author, price):*

*self.title = title*

*self.quantity = quantity*

*self.author = author*

*self.price = price*

*def \_\_repr\_\_(self):*

*return f"Book: {self.title}, Quantity: {self.quantity}, Author: {self.author}, Price: {self.price}"*

*book1 = Book('Book 1', 12, 'Author 1', 120)*

*book2 = Book('Book 2', 18, 'Author 2', 220)*

*book3 = Book('Book 3', 28, 'Author 3', 320)*

*print(book1)*

*print(book2)*

*print(book3)*

**Output:**

Book: Book 1, Quantity: 12, Author: Author 1, Price: 120

Book: Book 2, Quantity: 18, Author: Author 2, Price: 220

Book: Book 3, Quantity: 28, Author: Author 3, Price: 320

**What is Encapsulation?**

**Encapsulation** is the process of **preventing clients from accessing certain properties**, which can only be accessed through specific methods.

Private attributes are inaccessible attributes, and information hiding is the process of making particular attributes private. You use two underscores to declare private characteristics.

Let's introduce a private attribute called \_\_discount in the Book class.

*class Book:*

*def \_\_init\_\_(self, title, quantity, author, price):*

*self.title = title*

*self.quantity = quantity*

*self.author = author*

*self.price = price*

*self.\_\_discount = 0.10*

*def \_\_repr\_\_(self):*

*return f"Book: {self.title}, Quantity: {self.quantity}, Author: {self.author}, Price: {self.price}"*

*book1 = Book('Book 1', 12, 'Author 1', 120)*

*print(book1.title)*

*print(book1.quantity)*

*print(book1.author)*

*print(book1.price)*

*print(book1.\_\_discount)*

**Output:**

Book 1

12

Author 1

120

Traceback (most recent call last):

File "C:\Users\ashut\Desktop\Test\hello\test.py", line 19, in <module>

print(book1.\_\_discount)

AttributeError: 'Book' object has no attribute '\_\_discount'

We can see that all the attributes are printed except the private attribute \_\_discount. You use **getter and setter methods to access private attributes**.

We make the price property private in the following code example, and we use a setter method to assign the discount attribute and a getter function to get the price attribute.

*class Book:*

*def \_\_init\_\_(self, title, quantity, author, price):*

*self.title = title*

*self.quantity = quantity*

*self.author = author*

*self.\_\_price = price*

*self.\_\_discount = None*

*def set\_discount(self, discount):*

*self.\_\_discount = discount*

*def get\_price(self):*

*if self.\_\_discount:*

*return self.\_\_price \* (1-self.\_\_discount)*

*return self.\_\_price*

*def \_\_repr\_\_(self):*

*return f"Book: {self.title}, Quantity: {self.quantity}, Author: {self.author}, Price: {self.get\_price()}"*

This time we'll create two objects, one for the purchase of single book and another for the purchase of books in bulk quantity. While purchasing books in bulk quantity, we get a discount of 20%, so we'll use the set\_discount() method to set the discount to 20% in that case.

*single\_book = Book('Two States', 1, 'Chetan Bhagat', 200)*

*bulk\_books = Book('Two States', 25, 'Chetan Bhagat', 200)*

*bulk\_books.set\_discount(0.20)*

*print(single\_book.get\_price())*

*print(bulk\_books.get\_price())*

*print(single\_book)*

*print(bulk\_books)*

Output:

*200*

*160.0*

*Book: Two States, Quantity: 1, Author: Chetan Bhagat, Price: 200*

*Book: Two States, Quantity: 25, Author: Chetan Bhagat, Price: 160.0*

**What is Inheritance?**

**Inheritance** is regarded as the most significant characteristics of OOP. **A class's ability to inherit methods and/or characteristics** from another class is known as inheritance.

**The subclass or child class** is the class that inherits. **The superclass or parent class** is the class from which methods and/or attributes are inherited.

Two new classes have been added to our bookseller's sales software: a Novel class and Academic class.

We can see that regardless of whether a book is classified as novel or academic, it may have some similar attributes like title and author, as well as common methods like get\_price() and set\_discount(). Rewriting all that code for each new class is a waste of time, effort, and memory.

*class Book:*

*def \_\_init\_\_(self, title, quantity, author, price):*

*self.title = title*

*self.quantity = quantity*

*self.author = author*

*self.\_\_price = price*

*self.\_\_discount = None*

*def set\_discount(self, discount):*

*self.\_\_discount = discount*

*def get\_price(self):*

*if self.\_\_discount:*

*return self.\_\_price \* (1-self.\_\_discount)*

*return self.\_\_price*

*def \_\_repr\_\_(self):*

*return f"Book: {self.title}, Quantity: {self.quantity}, Author: {self.author}, Price: {self.get\_price()}"*

*class Novel(Book):*

*def \_\_init\_\_(self, title, quantity, author, price, pages):*

*super().\_\_init\_\_(title, quantity, author, price)*

*self.pages = pages*

*class Academic(Book):*

*def \_\_init\_\_(self, title, quantity, author, price, branch):*

*super().\_\_init\_\_(title, quantity, author, price)*

*self.branch = branch*

Let's create objects for these classes to visualize them.

*novel1 = Novel('Two States', 20, 'Chetan Bhagat', 200, 187)*

*novel1.set\_discount(0.20)*

*academic1 = Academic('Python Foundations', 12, 'PSF', 655, 'IT')*

*print(novel1)*

*print(academic1)*

Output:

Book: Two States, Quantity: 20, Author: Chetan Bhagat, Price: 160.0

Book: Python Foundations, Quantity: 12, Author: PSF, Price: 655

**What is Polymorphism?**

The term 'polymorphism' comes from the Greek language and means 'something that takes on multiple forms.'

**Polymorphism** refers to a subclass's **ability to adapt a method that already exists in its superclass to meet its needs**. To put it another way, a subclass can use a method from its superclass as is or modify it as needed.

*class Academic(Book):*

*def \_\_init\_\_(self, title, quantity, author, price, branch):*

*super().\_\_init\_\_(title, quantity, author, price)*

*self.branch = branch*

*def \_\_repr\_\_(self):*

*return f"Book: {self.title}, Branch: {self.branch}, Quantity: {self.quantity}, Author: {self.author}, Price: {self.get\_price()}"*

The Book superclass has a specific method called *\_\_repr\_\_.* This method can be used by subclass Novel so that it is called whenever an object is printed.

The Academic subclass, on the other hand, is defined with its own *\_\_repr\_\_* special function in the example code above. The Academic subclass will invoke its own method by suppressing the same method present in its superclass, thanks to polymorphism.

*novel1 = Novel('Two States', 20, 'Chetan Bhagat', 200, 187)*

*novel1.set\_discount(0.20)*

*academic1 = Academic('Python Foundations', 12, 'PSF', 655, 'IT')*

*print(novel1)*

*print(academic1)*

**Output**:

Book: Two States, Quantity: 20, Author: Chetan Bhagat, Price: 160.0

Book: Python Foundations, Branch: IT, Quantity: 12, Author: PSF, Price: 655

**What is Abstraction?**

**Abstraction isn't supported directly in Python**. Calling a magic method, on the other hand, allows for abstraction.

If an abstract method is declared in a superclass, subclasses that inherit from the superclass must have their own implementation of the method.

A superclass's abstract method will never be called by its subclasses. But the abstraction aids in the maintenance of a similar structure across all subclasses.

In our parent class Book, we have defined the *\_\_repr\_\_* method. Let's make that method abstract, forcing every subclass to compulsorily have its own *\_\_repr\_\_* method.

*from abc import ABC, abstractmethod*

*class Book(ABC):*

*def \_\_init\_\_(self, title, quantity, author, price):*

*self.title = title*

*self.quantity = quantity*

*self.author = author*

*self.\_\_price = price*

*self.\_\_discount = None*

*def set\_discount(self, discount):*

*self.\_\_discount = discount*

*def get\_price(self):*

*if self.\_\_discount:*

*return self.\_\_price \* (1-self.\_\_discount)*

*return self.\_\_price*

*@abstractmethod*

*def \_\_repr\_\_(self):*

*return f"Book: {self.title}, Quantity: {self.quantity}, Author: {self.author}, Price: {self.get\_price()}"*

*class Novel(Book):*

*def \_\_init\_\_(self, title, quantity, author, price, pages):*

*super().\_\_init\_\_(title, quantity, author, price)*

*self.pages = pages*

*class Academic(Book):*

*def \_\_init\_\_(self, title, quantity, author, price, branch):*

*super().\_\_init\_\_(title, quantity, author, price)*

*self.branch = branch*

*def \_\_repr\_\_(self):*

*return f"Book: {self.title}, Branch: {self.branch}, Quantity: {self.quantity}, Author: {self.author}, Price: {self.get\_price()}"*

*novel1 = Novel('Two States', 20, 'Chetan Bhagat', 200, 187)*

*novel1.set\_discount(0.20)*

*academic1 = Academic('Python Foundations', 12, 'PSF', 655, 'IT')*

*print(novel1)*

*print(academic1)*

In the above code, we have inherited the ABC class to create the Book class. We've made the *\_\_repr\_\_* method abstract by adding the *@abstractmethod* decorator.

While creating the Novel class, we intentionally missed the implementation of the *\_\_repr\_\_* method to see what happens.

**Output:**

Traceback (most recent call last):

File "C:\Users\ashut\Desktop\Test\hello\test.py", line 40, in <module>

novel1 = Novel('Two States', 20, 'Chetan Bhagat', 200, 187)

TypeError: Can't instantiate abstract class Novel with abstract method \_\_repr\_\_

We get a TypeError saying we cannot instantiate object of the Novel class. Let's add the implementation of the \_\_repr\_\_ method and see what happens now.

*class Novel(Book):*

*def \_\_init\_\_(self, title, quantity, author, price, pages):*

*super().\_\_init\_\_(title, quantity, author, price)*

*self.pages = pages*

*def \_\_repr\_\_(self):*

*return f"Book: {self.title}, Quantity: {self.quantity}, Author: {self.author}, Price: {self.get\_price()}"*

Output:

Book: Two States, Quantity: 20, Author: Chetan Bhagat, Price: 160.0

Book: Python Foundations, Branch: IT, Quantity: 12, Author: PSF, Price: 655

Now it works fine.

**Method Overloading**

The concept of method overloading is found in almost every well-known programming language that follows object-oriented programming concepts. It simply refers to **the use of many methods with the same name that take various numbers of arguments within a single class.**

Let's now understand method overloading with the help of the following code:

*class OverloadingDemo:*

*def add(self, x, y):*

*print(x+y)*

*def add(self, x, y, z):*

*print(x+y+z)*

*obj = OverloadingDemo()*

*obj.add(2, 3)*

**Output:**

Traceback (most recent call last):

File "C:\Users\ashut\Desktop\Test\hello\setup.py", line 10, in <module>

obj.add(2, 3)

TypeError: add() missing 1 required positional argument: 'z'

You're probably wondering why this happened. As a result, the error is displayed because Python only remembers the most recent definition of add(self, x, y, z), which takes three parameters in addition to *self*. As a result, three arguments must be passed to the add() method when it is called. **To put it another way, it disregards the prior definition of add().**

**Thus, Python doesn't support Method Overloading by default.**

**Method Overriding**

When a method with the same name and arguments is used in both a derived class and a base or super class, we say that the derived class method "**overrides**" the method provided in the base class.

When the overridden method gets called, the derived class's method is always invoked. The method that was used in the base class is now hidden.

*class ParentClass:*

*def call\_me(self):*

*print("I am parent class")*

*class ChildClass(ParentClass):*

*def call\_me(self):*

*print("I am child class")*

*pobj = ParentClass()*

*pobj.call\_me()*

*cobj = ChildClass()*

*cobj.call\_me()*

**Output:**

I am parent class

I am child class

In the above code, when the call\_me() method was called on the child object, the call\_me() from the child class was invoked. We can invoke the parent class's call\_me() method from the child class using super(), like this:

*class ParentClass:*

*def call\_me(self):*

*print("I am parent class")*

*class ChildClass(ParentClass):*

*def call\_me(self):*

*print("I am child class")*

*super().call\_me()*

*pobj = ParentClass()*

*pobj.call\_me()*

*cobj = ChildClass()*

*cobj.call\_me()*

Output:

I am parent class

I am child class

I am parent class

## What is a function of “\_init” in OOP?

The *\_\_init\_\_* special method, also known as a **Constructor**, is used to initialize the Book class with attributes such as title, quantity, author, and price.

## What is “self” in OOP?

*book1, book2 and book3* are distinct objects of the class Book. The term *self* in the attributes refers to the corresponding instances (objects).

*print(book1)*

*print(book2)*

*print(book3)*

*Output:*

*<\_\_main\_\_.Book object at 0x00000156EE59A9D0>*

*<\_\_main\_\_.Book object at 0x00000156EE59A8B0>*

*<\_\_main\_\_.Book object at 0x00000156EE59ADF0>*

## What is a decorator?

In Python, decorators are essential functions. These add extension/ functionality to an existing function without altering the structure of the function itself. Instead, decorators take another function as their argument and return yet another function. This, too, is a common Python data structure interview question.

## Github/Git commit/push

**1. What is Git?**

Git is a version control system for tracking changes in computer files and is used to help coordinate work among several people on a project while tracking progress over time. In other words, it’s a tool that facilitates source code management in software development.

Git favors both programmers and non-technical users by keeping track of their project files. It enables multiple users to work together and handles large projects efficiently.

**2. What do you understand by the term ‘Version Control System’?**

A version control system (VCS) records all the changes made to a file or set of data, so a specific version may be called later if needed.

This helps ensure that all team members are working on the latest version of the file server.

**6. What benefits come with using GIT?**

* Data replication and redundancy are both possible.
* It is a service with high availability.
* There can only be one Git directory per repository.
* Excellent network and disc performance are achieved.
* On any project, collaboration is very simple.

**7. What’s the difference between**[**Git and GitHub**](https://www.simplilearn.com/tutorials/git-tutorial/git-vs-github)**?**

|  |  |
| --- | --- |
| **Git** | **GitHub** |
| Git is a software | GitHub is a service |
| [Git can be installed](https://www.simplilearn.com/tutorials/git-tutorial/git-installation-on-windows) locally on the system | GitHub is hosted on the web |
| Provides a desktop interface called git GUI | Provides a desktop interface called GitHub Desktop. |
| It does not support user management features | Provides built-in user management |

**9. How can you initialize a repository in Git?**

If you want to initialize an empty repository to a directory in Git, you need to enter the git init command. After this command, a hidden .git folder will appear.



**11. Name a few Git commands with their function.**

* Git config - Configure the username and email address
* Git add - Add one or more files to the staging area
* Git diff - View the changes made to the file
* Git init - Initialize an empty Git repository
* Git commit - Commit changes to head but not to the remote repository

**12. What are the advantages of using Git?**

* Faster release cycles
* Easy team collaboration
* Widespread acceptance
* Maintains the integrity of source code
* [Pull requests](https://www.simplilearn.com/tutorials/git-tutorial/git-pull-request)

**14. What is the correct syntax to add a message to a commit?**

 git commit -m "x files created"

**15. Which command is used to create an empty Git repository?**

git init - This [command](https://www.simplilearn.com/tutorials/git-tutorial/git-commands) helps to create an empty repository while working on a project.

**16. What does git pull origin master do?**

The git pull origin master fetches all the changes from the master branch onto the origin and integrates them into the local branch.

git pull = git fetch + git merge origin/ master

After having gone through the beginner level Git interview questions, let us now look at intermediate GIT interview questions and answers.

**17.  What does the git push command do?**

The [Git push command](https://www.simplilearn.com/tutorials/git-tutorial/git-push-command) is used to push the content in a local repository to a remote repository. After a local repository has been modified, a push is executed to share the modifications with remote team members.



**18. Difference between git fetch and git pull.**

|  |  |
| --- | --- |
| **Git Fetch** | **Git Pull** |
| The Git fetch command only downloads new data from a remote repository. | Git pull updates the current HEAD branch with the latest changes from the remote server. |
| It does not integrate any of these new data into your working files. | Downloads new data and integrate it with the current working files. |
| Command - git fetch origin  git fetch --all | Tries to merge remote changes with your local ones.  Command - git pull origin master |

**19. GitHub, GitLab and Bitbucket are examples of git repository \_\_\_\_\_\_\_ function?**

hosting. All the three are services for hosting Git repositories

**20. What do you understand about the Git merge conflict?**

A [Git merge conflict](https://www.simplilearn.com/tutorials/git-tutorial/merge-conflicts-in-git) is an event that occurs when Git is unable to resolve the differences in code between the two commits automatically.

Git is capable of automatically merging the changes only if the commits are on different lines or branches.



**21. How do you resolve conflicts in Git?**

Here are the steps that will help you resolve conflicts in Git:

* Identify the files responsible for the conflicts.
* Implement the desired changes to the files
* Add the files using the git add command.
* The last step is to commit the changes in the file with the help of the git commit command.

## Multithreading

Running several threads is similar to running several different programs concurrently, but with the following benefits −

* Multiple threads within a process share the same data space with the main thread and can therefore share information or communicate with each other more easily than if they were separate processes.
* Threads sometimes called light-weight processes and they do not require much memory overhead; they are cheaper than processes.

A thread has a beginning, an execution sequence, and a conclusion. It has an instruction pointer that keeps track of where within its context it is currently running.

* It can be pre-empted (interrupted)
* It can temporarily be put on hold (also known as sleeping) while other threads are running - this is called yielding.

**Starting a New Thread**

To spawn another thread, you need to call following method available in *thread* module −

thread.start\_new\_thread ( function, args[, kwargs] )

This method call enables a fast and efficient way to create new threads in both Linux and Windows.

The method call returns immediately and the child thread starts and calls function with the passed list of *args*. When function returns, the thread terminates.

Here, *args* is a tuple of arguments; use an empty tuple to call function without passing any arguments. *kwargs*is an optional dictionary of keyword arguments.

**Example**

#!/usr/bin/python

import thread

import time

# Define a function for the thread

def print\_time( threadName, delay):

count = 0

while count < 5:

time.sleep(delay)

count += 1

print "%s: %s" % ( threadName, time.ctime(time.time()) )

# Create two threads as follows

try:

thread.start\_new\_thread( print\_time, ("Thread-1", 2, ) )

thread.start\_new\_thread( print\_time, ("Thread-2", 4, ) )

except:

print "Error: unable to start thread"

while 1:

pass

When the above code is executed, it produces the following result −

Thread-1: Thu Jan 22 15:42:17 2009

Thread-1: Thu Jan 22 15:42:19 2009

Thread-2: Thu Jan 22 15:42:19 2009

Thread-1: Thu Jan 22 15:42:21 2009

Thread-2: Thu Jan 22 15:42:23 2009

Thread-1: Thu Jan 22 15:42:23 2009

Thread-1: Thu Jan 22 15:42:25 2009

Thread-2: Thu Jan 22 15:42:27 2009

Thread-2: Thu Jan 22 15:42:31 2009

Thread-2: Thu Jan 22 15:42:35 2009

Although it is very effective for low-level threading, but the *thread* module is very limited compared to the newer threading module.

AD

**The Threading Module**

The newer threading module included with Python 2.4 provides much more powerful, high-level support for threads than the thread module discussed in the previous section.

The *threading* module exposes all the methods of the *thread* module and provides some additional methods −

* **threading.activeCount()** − Returns the number of thread objects that are active.
* **threading.currentThread()** − Returns the number of thread objects in the caller's thread control.
* **threading.enumerate()** − Returns a list of all thread objects that are currently active.

In addition to the methods, the threading module has the *Thread* class that implements threading. The methods provided by the *Thread* class are as follows −

* **run()** − The run() method is the entry point for a thread.
* **start()** − The start() method starts a thread by calling the run method.
* **join([time])** − The join() waits for threads to terminate.
* **isAlive()** − The isAlive() method checks whether a thread is still executing.
* **getName()** − The getName() method returns the name of a thread.
* **setName()** − The setName() method sets the name of a thread.

**Creating Thread Using Threading Module**

To implement a new thread using the threading module, you have to do the following −

* Define a new subclass of the *Thread* class.
* Override the *\_\_init\_\_(self [,args])* method to add additional arguments.
* Then, override the run(self [,args]) method to implement what the thread should do when started.

Once you have created the new *Thread* subclass, you can create an instance of it and then start a new thread by invoking the *start()*, which in turn calls *run()* method.

**Example**

#!/usr/bin/python

import threading

import time

exitFlag = 0

class myThread (threading.Thread):

def \_\_init\_\_(self, threadID, name, counter):

threading.Thread.\_\_init\_\_(self)

self.threadID = threadID

self.name = name

self.counter = counter

def run(self):

print "Starting " + self.name

print\_time(self.name, 5, self.counter)

print "Exiting " + self.name

def print\_time(threadName, counter, delay):

while counter:

if exitFlag:

threadName.exit()

time.sleep(delay)

print "%s: %s" % (threadName, time.ctime(time.time()))

counter -= 1

# Create new threads

thread1 = myThread(1, "Thread-1", 1)

thread2 = myThread(2, "Thread-2", 2)

# Start new Threads

thread1.start()

thread2.start()

print "Exiting Main Thread"

When the above code is executed, it produces the following result −

Starting Thread-1

Starting Thread-2

Exiting Main Thread

Thread-1: Thu Mar 21 09:10:03 2013

Thread-1: Thu Mar 21 09:10:04 2013

Thread-2: Thu Mar 21 09:10:04 2013

Thread-1: Thu Mar 21 09:10:05 2013

Thread-1: Thu Mar 21 09:10:06 2013

Thread-2: Thu Mar 21 09:10:06 2013

Thread-1: Thu Mar 21 09:10:07 2013

Exiting Thread-1

Thread-2: Thu Mar 21 09:10:08 2013

Thread-2: Thu Mar 21 09:10:10 2013

Thread-2: Thu Mar 21 09:10:12 2013

Exiting Thread-2

AD

**Synchronizing Threads**

The threading module provided with Python includes a simple-to-implement locking mechanism that allows you to synchronize threads. A new lock is created by calling the *Lock()* method, which returns the new lock.

The *acquire(blocking)* method of the new lock object is used to force threads to run synchronously. The optional *blocking* parameter enables you to control whether the thread waits to acquire the lock.

If *blocking* is set to 0, the thread returns immediately with a 0 value if the lock cannot be acquired and with a 1 if the lock was acquired. If blocking is set to 1, the thread blocks and wait for the lock to be released.

The *release()* method of the new lock object is used to release the lock when it is no longer required.

**Example**

#!/usr/bin/python

import threading

import time

class myThread (threading.Thread):

def \_\_init\_\_(self, threadID, name, counter):

threading.Thread.\_\_init\_\_(self)

self.threadID = threadID

self.name = name

self.counter = counter

def run(self):

print "Starting " + self.name

# Get lock to synchronize threads

threadLock.acquire()

print\_time(self.name, self.counter, 3)

# Free lock to release next thread

threadLock.release()

def print\_time(threadName, delay, counter):

while counter:

time.sleep(delay)

print "%s: %s" % (threadName, time.ctime(time.time()))

counter -= 1

threadLock = threading.Lock()

threads = []

# Create new threads

thread1 = myThread(1, "Thread-1", 1)

thread2 = myThread(2, "Thread-2", 2)

# Start new Threads

thread1.start()

thread2.start()

# Add threads to thread list

threads.append(thread1)

threads.append(thread2)

# Wait for all threads to complete

for t in threads:

t.join()

print "Exiting Main Thread"

When the above code is executed, it produces the following result −

Starting Thread-1

Starting Thread-2

Thread-1: Thu Mar 21 09:11:28 2013

Thread-1: Thu Mar 21 09:11:29 2013

Thread-1: Thu Mar 21 09:11:30 2013

Thread-2: Thu Mar 21 09:11:32 2013

Thread-2: Thu Mar 21 09:11:34 2013

Thread-2: Thu Mar 21 09:11:36 2013

Exiting Main Thread

**Multithreaded Priority Queue**

The *Queue* module allows you to create a new queue object that can hold a specific number of items. There are following methods to control the Queue −

* **get()** − The get() removes and returns an item from the queue.
* **put()** − The put adds item to a queue.
* **qsize()** − The qsize() returns the number of items that are currently in the queue.
* **empty()** − The empty( ) returns True if queue is empty; otherwise, False.
* **full()** − the full() returns True if queue is full; otherwise, False.

**Example**

#!/usr/bin/python

import Queue

import threading

import time

exitFlag = 0

class myThread (threading.Thread):

def \_\_init\_\_(self, threadID, name, q):

threading.Thread.\_\_init\_\_(self)

self.threadID = threadID

self.name = name

self.q = q

def run(self):

print "Starting " + self.name

process\_data(self.name, self.q)

print "Exiting " + self.name

def process\_data(threadName, q):

while not exitFlag:

queueLock.acquire()

if not workQueue.empty():

data = q.get()

queueLock.release()

print "%s processing %s" % (threadName, data)

else:

queueLock.release()

time.sleep(1)

threadList = ["Thread-1", "Thread-2", "Thread-3"]

nameList = ["One", "Two", "Three", "Four", "Five"]

queueLock = threading.Lock()

workQueue = Queue.Queue(10)

threads = []

threadID = 1

# Create new threads

for tName in threadList:

thread = myThread(threadID, tName, workQueue)

thread.start()

threads.append(thread)

threadID += 1

# Fill the queue

queueLock.acquire()

for word in nameList:

workQueue.put(word)

queueLock.release()

# Wait for queue to empty

while not workQueue.empty():

pass

# Notify threads it's time to exit

exitFlag = 1

# Wait for all threads to complete

for t in threads:

t.join()

print "Exiting Main Thread"

When the above code is executed, it produces the following result −

Starting Thread-1

Starting Thread-2

Starting Thread-3

Thread-1 processing One

Thread-2 processing Two

Thread-3 processing Three

Thread-1 processing Four

Thread-2 processing Five

Exiting Thread-3

Exiting Thread-1

Exiting Thread-2

Exiting Main Thread

## Explain what zip function performs

## Explain what map function performs

## Explain what lambda function performs

## What are the differences between args and kwargs?

## Python – JPM questions

1. Bubble sort – co to jest?
2. Projekty: przykłady
3. Struktury danych – wymień
4. Różnica między queue a stack … ?
5. Lambda function
6. Jak zbudować klasę w Pythonie?
7. Jak zainicjować klasę?
8. Co robi self?
9. Czym jest decorator?
10. Co to jest list comprehension?
11. Co się stanie jak weźmiesz z array [-1] indeks?
12. Chyba coś o zip function…?
13. Właściwości Pythona jako języka OOP …? Co to jest inheritance?
14. Pytanie o Git. Jak stworzyć branch? Komenda
15. Multithread vs multiprocess

<https://data-flair.training/blogs/top-python-interview-questions-answer/>

# ****SQL****

### **What is the difference between SQL and MySQL?**

|  |  |
| --- | --- |
| **SQL vs MySQL** | |
| **SQL** | **MySQL** |
| SQL is a standard language which stands for Structured Query Language based on the English language | MySQL is a database management system. |
| SQL is the core of the relational database which is used for accessing and managing database | MySQL is an RDMS (Relational Database Management System) such as SQL Server, Informix etc. |

### **What are the different subsets of SQL?**

* Data Definition Language (DDL) – It allows you to perform various operations on the database such as CREATE, ALTER, and DELETE objects.
* Data Manipulation Language(DML) – It allows you to access and manipulate data. It helps you to insert, update, delete and retrieve data from the database.
* Data Control Language(DCL) – It allows you to control access to the database. Example – Grant, Revoke access permissions.

### **What do you mean by DBMS? What are its different types?**

A [**Database Management System**](https://www.edureka.co/blog/dbms-tutorial/) (**DBMS**) is a  software application that interacts with the user, applications, and the database itself to capture and analyze data. A database is a structured collection of data.

A DBMS allows a user to interact with the database. The data stored in the database can be modified, retrieved and deleted and can be of any type like strings, numbers, images, etc.

There are two types of DBMS:

* Relational Database Management System: The data is stored in relations (tables). Example – MySQL.
* Non-Relational Database Management System: There is no concept of relations, tuples and attributes.  Example – MongoDB

Let’s move to the next question in this SQL Interview Questions.

### **What is RDBMS? How is it different from DBMS?**

A relational database management system (RDBMS) is a set of applications and features that allow IT professionals and others to develop, edit, administer, and interact with relational databases. Most commercial relational database management systems use Structured Query Language (SQL) to access the database, which is stored in the form of tables.  
The RDBMS is the most widely used database system in businesses all over the world. It offers a stable means of storing and retrieving massive amounts of data.

Databases, in general, hold collections of data that may be accessed and used in other applications. The development, administration, and use of database platforms are all supported by a database management system.

A relational database management system (RDBMS) is a type of database management system (DBMS) that stores data in a row-based table structure that links related data components. An RDBMS contains functions that ensure the data’s security, accuracy, integrity, and consistency. This is not the same as the file storage utilized by a database management system.

The following are some further distinctions between database management systems and relational database management systems:

**The number of users who are permitted to utilise the system**  
A DBMS can only handle one user at a time, whereas an RDBMS can handle numerous users.  
**Hardware and software specifications**  
In comparison to an RDBMS, a DBMS requires fewer software and hardware.  
**Amount of information**  
RDBMSes can handle any quantity of data, from tiny to enormous, whereas DBMSes are limited to small amounts.  
**The structure of the database**  
Data is stored in a hierarchical format in a DBMS, whereas an RDBMS uses a table with headers that serve as column names and rows that hold the associated values.  
**Implementation of the ACID principle**  
The atomicity, consistency, isolation, and durability (ACID) concept is not used by DBMSs for data storage. RDBMSes, on the other hand, use the ACID model to organize their data and assure consistency.  
**Databases that are distributed**  
A DBMS will not provide complete support for distributed databases, whereas an RDBMS will.  
**Programs that are managed**  
A DBMS focuses on keeping databases that are present within the computer network and system hard discs, whereas an RDBMS helps manage relationships between its incorporated tables of data.  
**Normalization of databases is supported**  
A RDBMS can be normalized , but a DBMS cannot be normalized.

### What is a Self-Join?

A self-join is a type of join that can be used to connect two tables. As a result, it is a unary relationship. Each row of the table is attached to itself and all other rows of the same table in a self-join. As a result, a self-join is mostly used to combine and compare rows from the same database table.

### What is the SELECT statement?

A SELECT command gets zero or more rows from one or more database tables or views. The most frequent data manipulation language (DML) command is SELECT in most applications. SELECT queries define a result set, but not how to calculate it, because SQL is a declarative programming language.

### What are some common clauses used with SELECT query in SQL?

The following are some frequent SQL clauses used in conjunction with a SELECT query:

**WHERE** clause: In SQL, the WHERE clause is used to filter records that are required depending on certain criteria.  
**ORDER BY** clause: The ORDER BY clause in SQL is used to sort data in ascending (ASC) or descending (DESC) order depending on specified field(s) (DESC).  
**GROUP BY** clause: GROUP BY clause in SQL is used to group entries with identical data and may be used with aggregation methods to obtain summarised database results.  
**HAVING** clause in SQL is used to filter records in combination with the GROUP BY clause. It is different from WHERE, since the WHERE clause cannot filter aggregated records.

### What are UNION, MINUS and INTERSECT commands?

The UNION operator is used to combine the results of two tables while also removing duplicate entries.  
The MINUS operator is used to return rows from the first query but not from the second query.  
The INTERSECT operator is used to combine the results of both queries into a single row.  
Before running either of the above SQL statements, certain requirements must be satisfied –  
Within the clause, each SELECT query must have the same amount of columns.  
The data types in the columns must also be comparable.  
In each SELECT statement, the columns must be in the same order.

### What is Cursor? How to use a Cursor?

After any variable declaration, DECLARE a cursor. A SELECT Statement must always be coupled with the cursor definition.

To start the result set, move the cursor over it. Before obtaining rows from the result set, the OPEN statement must be executed.

To retrieve and go to the next row in the result set, use the FETCH command.

To disable the cursor, use the CLOSE command.

Finally, use the DEALLOCATE command to remove the cursor definition and free up the resources connected with it.

### List the different types of relationships in SQL.

**There are different types of relations in the database:**

**One-to-One** – This is a connection between two tables in which each record in one table corresponds to the maximum of one record in the other.

**One-to-Many and Many-to-One** – This is the most frequent connection, in which a record in one table is linked to several records in another.

**Many-to-Many** – This is used when defining a relationship that requires several instances on each sides.

**Self-Referencing Relationships** – When a table has to declare a connection with itself, this is the method to employ.

### How to create empty tables with the same structure as another table?

To create empty tables:  
Using the INTO operator to fetch the records of one table into a new table while setting a WHERE clause to false for all entries, it is possible to create empty tables with the same structure. As a result, SQL creates a new table with a duplicate structure to accept the fetched entries, but nothing is stored into the new table since the WHERE clause is active.

### What is PostgreSQL?

In 1986, a team lead by Computer Science Professor Michael Stonebraker created PostgreSQL under the name Postgres. It was created to aid developers in the development of enterprise-level applications by ensuring data integrity and fault tolerance in systems. PostgreSQL is an enterprise-level, versatile, resilient, open-source, object-relational database management system that supports variable workloads and concurrent users. The international developer community has constantly backed it. PostgreSQL has achieved significant appeal among developers because to its fault-tolerant characteristics.  
It’s a very reliable database management system, with more than two decades of community work to thank for its high levels of resiliency, integrity, and accuracy. Many online, mobile, geospatial, and analytics applications utilise PostgreSQL as their primary data storage or data warehouse.

### **What are SQL comments?**

SQL Comments are used to clarify portions of SQL statements and to prevent SQL statements from being executed. Comments are quite important in many programming languages. The comments are not supported by a Microsoft Access database. As a result, the Microsoft Access database is used in the examples in Mozilla Firefox and Microsoft Edge.  
Single Line Comments: It starts with two consecutive hyphens (–).  
Multi-line Comments: It starts with /\* and ends with \*/.

### What is the usage of the NVL() function?

You may use the NVL function to replace null values with a default value. The function returns the value of the second parameter if the first parameter is null. If the first parameter is anything other than null, it is left alone.

This function is used in Oracle, not in SQL and MySQL. Instead of NVL() function, MySQL have IFNULL() and SQL Server have ISNULL() function.

Let’s move to the next question in this SQL Interview Questions.

### Explain character-manipulation functions? Explains its different types in SQL.

Change, extract, and edit the character string using character manipulation routines. The function will do its action on the input strings and return the result when one or more characters and words are supplied into it.

The character manipulation functions in SQL are as follows:

A) CONCAT (joining two or more values): This function is used to join two or more values together. The second string is always appended to the end of the first string.

B) SUBSTR: This function returns a segment of a string from a given start point to a given endpoint.

C) LENGTH: This function returns the length of the string in numerical form, including blank spaces.

D) INSTR: This function calculates the precise numeric location of a character or word in a string.

E) LPAD: For right-justified values, it returns the padding of the left-side character value.

F) RPAD: For a left-justified value, it returns the padding of the right-side character value.

G) TRIM: This function removes all defined characters from the beginning, end, or both ends of a string. It also reduced the amount of wasted space.

H) REPLACE: This function replaces all instances of a word or a section of a string (substring) with the other string value specified.

### Write the SQL query to get the third maximum salary of an employee from a table named employees.

Employee table

|  |  |
| --- | --- |
| employee\_name | salary |
| A | 24000 |
| C | 34000 |
| D | 55000 |
| E | 75000 |
| F | 21000 |
| G | 40000 |
| H | 50000 |

SELECT \* FROM(

SELECT employee\_name, salary, DENSE\_RANK()

OVER(ORDER BY salary DESC)r FROM Employee)

WHERE r=&n;

To find 3rd highest salary set n = 3

### **Q20. What is the difference between the RANK() and DENSE\_RANK() functions?**

The RANK() function in the result set defines the rank of each row within your ordered partition. If both rows have the same rank, the next number in the ranking will be the previous rank plus a number of duplicates. If we have three records at rank 4, for example, the next level indicated is 7.

The DENSE\_RANK() function assigns a distinct rank to each row within a partition based on the provided column value, with no gaps. It always indicates a ranking in order of precedence. This function will assign the same rank to the two rows if they have the same rank, with the next rank being the next consecutive number. If we have three records at rank 4, for example, the next level indicated is 5.

**Q21. What are Tables and Fields?**

A table is a collection of data components organized in rows and columns in a relational database. A table can also be thought of as a useful representation of relationships. The most basic form of data storage is the table. An example of an Employee table is shown below.

|  |  |  |  |
| --- | --- | --- | --- |
| ID | Name | Department | Salary |
| 1 | Rahul | Sales | 24000 |
| 2 | Rohini | Marketing | 34000 |
| 3 | Shylesh | Sales | 24000 |
| 4 | Tarun | Analytics | 30000 |

A Record or Row is a single entry in a table. In a table, a record represents a collection of connected data. The Employee table, for example, has four records.

A table is made up of numerous records (rows), each of which can be split down into smaller units called Fields(columns). ID, Name, Department, and Salary are the four fields in the Employee table above.

### What is a UNIQUE constraint?

The UNIQUE Constraint prevents identical values in a column from appearing in two records. The UNIQUE constraint guarantees that every value in a column is unique.

### What is a Self-Join?

A self-join is a type of join that can be used to connect two tables. As a result, it is a unary relationship. Each row of the table is attached to itself and all other rows of the same table in a self-join. As a result, a self-join is mostly used to combine and compare rows from the same database table.

### . What is the SELECT statement?

A SELECT command gets zero or more rows from one or more database tables or views. The most frequent data manipulation language (DML) command is SELECT in most applications. SELECT queries define a result set, but not how to calculate it, because SQL is a declarative programming language.

### . What are some common clauses used with SELECT query in SQL?

The following are some frequent SQL clauses used in conjunction with a SELECT query:

WHERE clause: In SQL, the WHERE clause is used to filter records that are required depending on certain criteria.  
ORDER BY clause: The ORDER BY clause in SQL is used to sort data in ascending (ASC) or descending (DESC) order depending on specified field(s) (DESC).  
GROUP BY clause: GROUP BY clause in SQL is used to group entries with identical data and may be used with aggregation methods to obtain summarised database results.  
HAVING clause in SQL is used to filter records in combination with the GROUP BY clause. It is different from WHERE, since the WHERE clause cannot filter aggregated records.

### . What are UNION, MINUS and INTERSECT commands?

**The UNION operator is used to combine the results of two tables while also removing duplicate entries.**

The MINUS operator is used to return rows from the first query but not from the second query.

The INTERSECT operator is used to combine the results of both queries into a single row.  
Before running either of the above SQL statements, certain requirements must be satisfied –

Within the clause, each SELECT query must have the same amount of columns.

The data types in the columns must also be comparable.

In each SELECT statement, the columns must be in the same order.

Let’s move to the next question in this SQL Interview Questions.

### . What is Cursor? How to use a Cursor?

After any variable declaration, DECLARE a cursor. A SELECT Statement must always be coupled with the cursor definition.

To start the result set, move the cursor over it. Before obtaining rows from the result set, the OPEN statement must be executed.

To retrieve and go to the next row in the result set, use the FETCH command.

To disable the cursor, use the CLOSE command.

Finally, use the DEALLOCATE command to remove the cursor definition and free up the resources connected with it.

### . List the different types of relationships in SQL.

There are different types of relations in the database:  
One-to-One – This is a connection between two tables in which each record in one table corresponds to the maximum of one record in the other.  
One-to-Many and Many-to-One – This is the most frequent connection, in which a record in one table is linked to several records in another.  
Many-to-Many – This is used when defining a relationship that requires several instances on each sides.  
Self-Referencing Relationships – When a table has to declare a connection with itself, this is the method to employ.

### **. What is SQL example?**

SQL is a database query language that allows you to edit, remove, and request data from databases. The following statements are a few examples of SQL statements:

* SELECT
* INSERT
* UPDATE
* DELETE
* CREATE DATABASE
* ALTER DATABASE

### . What are basic SQL skills?

SQL skills aid data analysts in the creation, maintenance, and retrieval of data from relational databases, which divide data into columns and rows. It also enables users to efficiently retrieve, update, manipulate, insert, and alter data.

The most fundamental abilities that a SQL expert should possess are:

1. Database Management
2. Structuring a Database
3. Creating SQL clauses and statements
4. SQL System SKills like MYSQL, PostgreSQL
5. PHP expertise is useful.
6. Analyze SQL data
7. Using WAMP with SQL to create a database
8. OLAP Skills

### What is schema in SQL Server?A schema is a visual representation of the database that is logical. It builds and specifies the relationships among the database’s numerous entities. It refers to the several kinds of constraints that may be applied to a database. It also describes the various data kinds. It may also be used on Tables and Views. Schemas come in a variety of shapes and sizes. Star schema and Snowflake schema are two of the most popular. The entities in a star schema are represented in a star form, whereas those in a snowflake schema are shown in a snowflake shape. Any database architecture is built on the foundation of schemas.

### How to create a temp table in SQL Server?

Temporary tables are created in TempDB and are erased automatically after the last connection is closed. We may use Temporary Tables to store and process interim results. When we need to store temporary data, temporary tables come in handy.  
  
The following is the syntax for creating a Temporary Table:  
  
CREATE TABLE #Employee (id INT, name VARCHAR(25))  
INSERT INTO #Employee VALUES (01, ‘Ashish’), (02, ‘Atul’)

Let’s move to the next question in this SQL Interview Questions.

### What is the case when in SQL Server?

The CASE statement is used to construct logic in which one column’s value is determined by the values of other columns.

At least one set of WHEN and THEN commands makes up the SQL Server CASE Statement. The condition to be tested is specified by the WHEN statement. If the WHEN condition returns TRUE, the THEN sentence explains what to do.

When none of the WHEN conditions return true, the ELSE statement is executed. The END keyword brings the CASE statement to a close.

|  |  |
| --- | --- |
| 1  2  3  4  5  6 | CASE  WHEN condition1 THEN result1  WHEN condition2 THEN result2  WHEN conditionN THEN resultN  ELSE result  END; |

### NoSQL vs SQL

In summary, the following are the five major distinctions between SQL and NoSQL:

Relational databases are SQL, while non-relational databases are NoSQL.

SQL databases have a specified schema and employ structured query language. For unstructured data, NoSQL databases use dynamic schemas.

SQL databases scale vertically, but NoSQL databases scale horizontally.

NoSQL databases are document, key-value, graph, or wide-column stores, whereas SQL databases are table-based.

SQL databases excel in multi-row transactions, while NoSQL excels at unstructured data such as documents and JSON.

Q36. What is the difference between NOW() and CURRENT\_DATE()?  
NOW() returns a constant time that indicates the time at which the statement began to execute. (Within a stored function or trigger, NOW() returns the time at which the function or triggering statement began to execute.  
The simple difference between NOW() and CURRENT\_DATE() is that NOW() will fetch the current date and time both in format ‘YYYY-MM\_DD HH:MM:SS’ while CURRENT\_DATE() will fetch the date of the current day ‘YYYY-MM\_DD’.

Let’s move to the next question in this SQL Interview Questions.

**Q37. What is BLOB and TEXT in MySQL?**

BLOB stands for Binary Huge Objects and can be used to store binary data, whereas TEXT may be used to store a large number of strings. BLOB may be used to store binary data, which includes images, movies, audio, and applications.  
BLOB values function similarly to byte strings, and they lack a character set. As a result, bytes’ numeric values are completely dependent on comparison and sorting.  
    TEXT values behave similarly to a character string or a non-binary string. The comparison/sorting of TEXT is completely dependent on the character set collection.

**Q38. How to remove duplicate rows in SQL?**

If the SQL table has duplicate rows, the duplicate rows must be removed.

Let’s assume the following table as our dataset:

|  |  |  |
| --- | --- | --- |
| ID | Name | Age |
| 1 | A | 21 |
| 2 | B | 23 |
| 2 | B | 23 |
| 4 | D | 22 |
| 5 | E | 25 |
| 6 | G | 26 |
| 5 | E | 25 |

The following SQL query removes the duplicate ids from the  table:  
  
DELETE FROM table WHERE ID IN (  
SELECT   
ID, COUNT(ID)   
FROM   table  
GROUP BY  ID  
HAVING   
COUNT (ID) > 1);

**Q39. How to create a stored procedure using SQL Server?**

A stored procedure is a piece of prepared SQL code that you can save and reuse again and over.  
So, if you have a SQL query that you create frequently, save it as a stored procedure and then call it to run it.  
You may also supply parameters to a stored procedure so that it can act based on the value(s) of the parameter(s) given.

Stored Procedure Syntax

CREATE PROCEDURE procedure\_name

AS

sql\_statement

GO;

Execute a Stored Procedure

EXEC procedure\_name;

**Q40. What is Database Black Box Testing?**

Black Box Testing is a software testing approach that involves testing the functions of software applications without knowing the internal code structure, implementation details, or internal routes. Black Box Testing is a type of software testing that focuses on the input and output of software applications and is totally driven by software requirements and specifications. Behavioral testing is another name for it.  
  
  
**Q41. What are the different types of SQL sandbox?**

### Databases Training

SQL Sandbox is a secure environment within SQL Server where untrusted programmes can be run. There are three different types of SQL sandboxes:

Safe Access Sandbox: In this environment, a user may execute SQL activities like as building stored procedures, triggers, and so on, but they can’t access the memory or create files.

Sandbox for External Access: Users can access files without having the ability to alter memory allocation.

Unsafe Access Sandbox: This contains untrustworthy code that allows a user to access memory.

Let’s move to the next question in this SQL Interview Questions.  
  
**Q43. How to find the nth highest salary in SQL?**  
The most typical interview question is to find the Nth highest pay in a table. This work can be accomplished using the dense rank() function.  
Employee table

|  |  |
| --- | --- |
| employee\_name | salary |
| A | 24000 |
| C | 34000 |
| D | 55000 |
| E | 75000 |
| F | 21000 |
| G | 40000 |
| H | 50000 |

SELECT \* FROM(

SELECT employee\_name, salary, DENSE\_RANK()

OVER(ORDER BY salary DESC)r FROM Employee)

WHERE r=&n;

To find to the 2nd highest salary set n = 2

To find 3rd highest salary set n = 3 and so on.

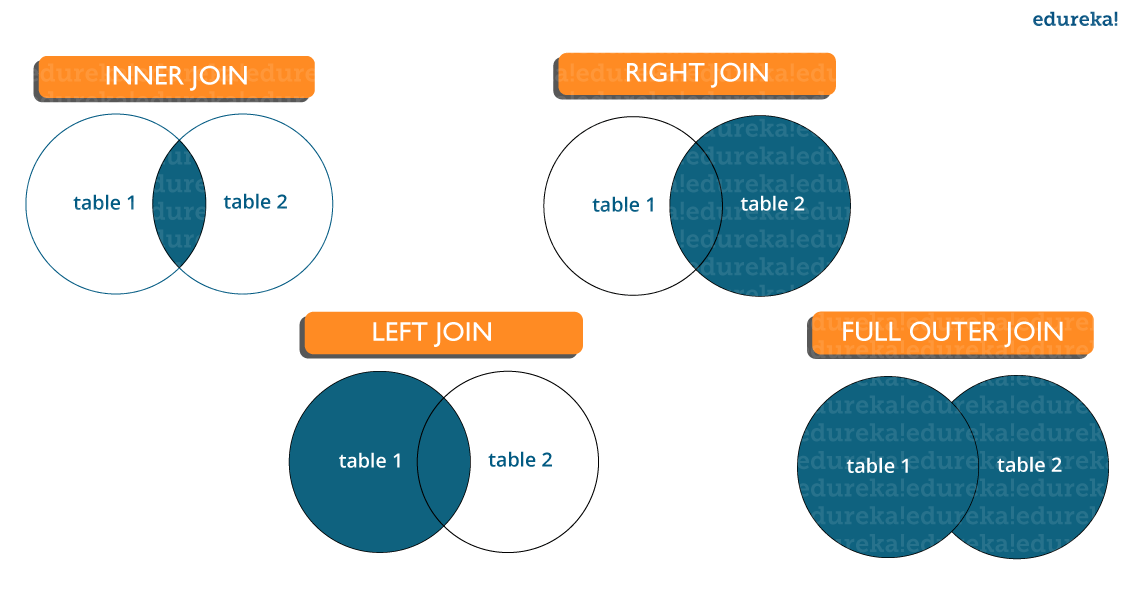
### **Q44. What do you mean by table and field in SQL?**

A table refers to a collection of data in an organised manner in form of rows and columns. A field refers to the number of columns in a table. For example:

***Table***: StudentInformation  
***Field***: Stu Id, Stu Name, Stu Marks

### **Q45. What are joins in SQL?**

A JOIN clause is used to combine rows from two or more tables, based on a related column between them. It is used to merge two tables or retrieve data from there. There are 4 types of joins, as you can refer to below:

****

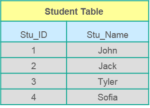
* **Inner join:** [Inner Join in SQL](https://www.edureka.co/blog/sql-joins-types) is the most common type of join. It is used to return all the rows from multiple tables where the join condition is satisfied.
* **Left Join:**  Left Join in SQL is used to return all the rows from the left table but only the matching rows from the right table where the join condition is fulfilled.
* **Right Join:** Right Join in SQL is used to return all the rows from the right table but only the matching rows from the left table where the join condition is fulfilled.
* **Full Join:** Full join returns all the records when there is a match in any of the tables. Therefore, it returns all the rows from the left-hand side table and all the rows from the right-hand side table.

Let’s move to the next question in this SQL Interview Questions.

### **Q46. What is the difference between CHAR and VARCHAR2 datatype in SQL?**

Both Char and Varchar2 are used for characters datatype but varchar2 is used for character strings of variable length whereas Char is used for strings of fixed length. For example, char(10) can only store 10 characters and will not be able to store a string of any other length whereas varchar2(10) can store any length i.e 6,8,2 in this variable.

### **Q47. What is a Primary key?**



* A[Primary key in SQL](https://www.edureka.co/blog/primary-key-in-sql/)is a column (or collection of columns) or a set of columns that uniquely identifies each row in the table.
* Uniquely identifies a single row in the table
* Null values not allowed

Example- In the Student table, Stu\_ID is the primary key.

### **Q48. What are Constraints?**

[Constraints in SQL](https://www.edureka.co/blog/sql-constraints/) are used to specify the limit on the data type of the table. It can be specified while creating or altering the table statement. The sample of constraints are:

* NOT NULL
* CHECK
* DEFAULT
* UNIQUE
* PRIMARY KEY
* FOREIGN KEY

### **Q49. What is the difference between DELETE and TRUNCATE statements?**

|  |  |
| --- | --- |
| **DELETE vs TRUNCATE** | |
| **DELETE** | **TRUNCATE** |
| Delete command is used to delete a row in a table. | Truncate is used to delete all the rows from a table. |
| You can rollback data after using delete statement. | You cannot rollback data. |
| It is a DML command. | It is a DDL command. |
| It is slower than truncate statement. | It is faster. |

### **Q50. What is a Unique key?**

* Uniquely identifies a single row in the table.
* Multiple values allowed per table.
* Null values allowed.

Apart from this SQL Interview Questions blog, if you want to get trained from professionals on this technology, you can opt for [structured training from edureka!](https://www.edureka.co/sql-essentials-training)

### **Q51. What is a Foreign key in SQL?**

* Foreign key maintains referential integrity by enforcing a link between the data in two tables.
* The foreign key in the child table references the primary key in the parent table.
* The [foreign key constraint](https://www.edureka.co/blog/foreign-key-sql/) prevents actions that would destroy links between the child and parent tables.

### **Q52. What do you mean by data integrity?**

Data Integrity defines the accuracy as well as the consistency of the data stored in a database. It also defines integrity constraints to enforce business rules on the data when it is entered into an application or a database.

### **Q53. What is the difference between clustered and non-clustered index in SQL?**

The differences between the clustered and non clustered index in SQL are :

1. Clustered index is used for easy retrieval of data from the database and its faster whereas reading from non clustered index is relatively slower.
2. Clustered index alters the way records are stored in a database as it sorts out rows by the column which is set to be clustered index whereas in a non clustered index, it does not alter the way it was stored but it creates a separate object within a table which points back to the original table rows after searching.
3. One table can only have one clustered index whereas it can have many non clustered index.

### **Q54. Write a SQL query to display the current date?**

In SQL, there is a built-in function called **GetDate()** which helps to return the current timestamp/date.

### **Q55.What do you understand by query optimization?**

The phase that identifies a plan for evaluation query which has the least estimated cost is known as query optimization.

The advantages of query optimization are as follows:

* The output is provided faster
* A larger number of queries can be executed in less time
* Reduces time and space complexity

### **Q56. What do you mean by Denormalization?**

Denormalization refers to a technique which is used to access data from higher to lower forms of a database. It helps the database managers to increase the performance of the entire infrastructure as it introduces redundancy into a table. It adds the redundant data into a table by incorporating database queries that combine data from various tables into a single table.

### **Q57. What are Entities and Relationships?**

**Entities**:  A person, place, or thing in the real world about which data can be stored in a database. Tables store data that represents one type of entity. For example – A bank database has a customer table to store customer information. The customer table stores this information as a set of attributes (columns within the table) for each customer.

**Relationships**: Relation or links between entities that have something to do with each other. For example – The customer name is related to the customer account number and contact information, which might be in the same table. There can also be relationships between separate tables (for example, customer to accounts).

Let’s move to the next question in this SQL Interview Questions.

### **What is an Index?**

An index refers to a performance tuning method of allowing faster retrieval of records from the table. An index creates an entry for each value and hence it will be faster to retrieve data.

### **Explain different types of index in SQL.**

There are three [types of index in SQL](https://www.edureka.co/blog/index-in-sql/) namely:

### **Unique Index:**

This index does not allow the field to have duplicate values if the column is unique indexed. If a primary key is defined, a unique index can be applied automatically.

### **Clustered Index:**

This index reorders the physical order of the table and searches based on the basis of key values. Each table can only have one clustered index.

### **Non-Clustered Index:**

Non-Clustered Index does not alter the physical order of the table and maintains a logical order of the data. Each table can have many nonclustered indexes.

### **Q60. What is Normalization and what are the advantages of it?**

[Normalization in SQL](https://www.edureka.co/blog/normalization-in-sql/) is the process of organizing data to avoid duplication and redundancy. Some of the advantages are:

* Better Database organization
* More Tables with smaller rows
* Efficient data access
* Greater Flexibility for Queries
* Quickly find the information
* Easier to implement Security
* Allows easy modification
* Reduction of redundant and duplicate data
* More Compact Database
* Ensure Consistent data after modification

Apart from this SQL Interview Questions Blog, if you want to get trained from professionals on this technology, you can opt for [structured training from edureka!](https://www.edureka.co/sql-essentials-training)

### **Q61. What is the difference between DROP and TRUNCATE commands?**

[DROP command](https://www.edureka.co/blog/sql-commands) removes a table and it cannot be rolled back from the database whereas TRUNCATE command removes all the rows from the table.

### **Q62. Explain different types of Normalization.**

There are many successive levels of normalization. These are called **normal forms**. Each consecutive normal form depends on the previous one.The first three normal forms are usually adequate.

Normal Forms are used in database tables to remove or decrease duplication. The following are the many forms:

**First Normal Form:**  
When every attribute in a relation is a single-valued attribute, it is said to be in first normal form. The first normal form is broken when a relation has a composite or multi-valued property.

**Second Normal Form:**

A relation is in second normal form if it meets the first normal form’s requirements and does not contain any partial dependencies. In 2NF, a relation has no partial dependence, which means it has no non-prime attribute that is dependent on any suitable subset of any table candidate key. Often, the problem may be solved by setting a single column Primary Key.

**Third Normal Form:**  
If a relation meets the requirements for the second normal form and there is no transitive dependency, it is said to be in the third normal form.

**Q63. What is OLTP?**

OLTP, or online transactional processing, allows huge groups of people to execute massive amounts of database transactions in real time, usually via the internet. A database transaction occurs when data in a database is changed, inserted, deleted, or queried.

What are the differences between OLTP and OLAP?

OLTP stands for online transaction processing, whereas OLAP stands for online analytical processing. OLTP is an online database modification system, whereas OLAP is an online database query response system.

**Q64. How to create empty tables with the same structure as another table?**

**To create empty tables:**

Using the INTO operator to fetch the records of one table into a new table while setting a WHERE clause to false for all entries, it is possible to create empty tables with the same structure. As a result, SQL creates a new table with a duplicate structure to accept the fetched entries, but nothing is stored into the new table since the WHERE clause is active.

**Q65. What is PostgreSQL?**

In 1986, a team lead by Computer Science Professor Michael Stonebraker created PostgreSQL under the name Postgres. It was created to aid developers in the development of enterprise-level applications by ensuring data integrity and fault tolerance in systems. PostgreSQL is an enterprise-level, versatile, resilient, open-source, object-relational database management system that supports variable workloads and concurrent users. The international developer community has constantly backed it. PostgreSQL has achieved significant appeal among developers because to its fault-tolerant characteristics.  
It’s a very reliable database management system, with more than two decades of community work to thank for its high levels of resiliency, integrity, and accuracy. Many online, mobile, geospatial, and analytics applications utilise PostgreSQL as their primary data storage or data warehouse.

**Q66. What are SQL comments?**

SQL Comments are used to clarify portions of SQL statements and to prevent SQL statements from being executed. Comments are quite important in many programming languages. The comments are not supported by a Microsoft Access database. As a result, the Microsoft Access database is used in the examples in Mozilla Firefox and Microsoft Edge.  
Single Line Comments: It starts with two consecutive hyphens (–).  
Multi-line Comments: It starts with /\* and ends with \*/.

Let’s move to the next question in this SQL Interview Questions.

**Q67. What is the difference between the RANK() and DENSE\_RANK() functions?**

The RANK() function in the result set defines the rank of each row within your ordered partition. If both rows have the same rank, the next number in the ranking will be the previous rank plus a number of duplicates. If we have three records at rank 4, for example, the next level indicated is 7.

[[](https://www.edureka.co/sql-essentials-training?utm_source=blogbanner&utm_campaign=batches)](https://www.edureka.co/sql-essentials-training?utm_source=blogbanner&utm_campaign=batches" \t "_blank)

### [SQL Essentials Training & Certification](https://www.edureka.co/sql-essentials-training?utm_source=blogbanner&utm_campaign=batches" \t "_blank)

[Weekday / Weekend BatchesSee Batch Details](https://www.edureka.co/sql-essentials-training?utm_source=blogbanner&utm_campaign=batches" \t "_blank)

The DENSE\_RANK() function assigns a distinct rank to each row within a partition based on the provided column value, with no gaps. It always indicates a ranking in order of precedence. This function will assign the same rank to the two rows if they have the same rank, with the next rank being the next consecutive number. If we have three records at rank 4, for example, the next level indicated is 5.

**Q68. What is SQL Injection?**

SQL injection is a sort of flaw in website and web app code that allows attackers to take control of back-end processes and access, retrieve, and delete sensitive data stored in databases. In this approach, malicious SQL statements are entered into a database entry field, and the database becomes exposed to an attacker once they are executed. By utilising data-driven apps, this strategy is widely utilised to get access to sensitive data and execute administrative tasks on databases. SQLi attack is another name for it.

The following are some examples of SQL injection:

* Getting access to secret data in order to change a SQL query to acquire the desired results.
* UNION attacks are designed to steal data from several database tables.
* Examine the database to get information about the database’s version and structure

**Q69. How many Aggregate functions are available in SQL?**

SQL aggregate functions provide information about a database’s data. AVG, for example, returns the average of a database column’s values.

SQL provides seven (7) aggregate functions, which are given below:

AVG(): returns the average value from specified columns.  
COUNT(): returns the number of table rows, including rows with null values.  
MAX(): returns the largest value among the group.  
MIN(): returns the smallest value among the group.  
SUM(): returns the total summed values(non-null) of the specified column.  
FIRST(): returns the first value of an expression.  
LAST(): returns the last value of an expression.

**Q70. What is the default ordering of data using the ORDER BY clause? How could it be changed?**

The ORDER BY clause in MySQL can be used without the ASC or DESC modifiers. The sort order is preset to ASC or ascending order when this attribute is absent from the ORDER BY clause.

**Q71. How do we use the DISTINCT statement? What is its use?**

The SQL DISTINCT keyword is combined with the SELECT query to remove all duplicate records and return only unique records. There may be times when a table has several duplicate records.  
The DISTINCT clause in SQL is used to eliminate duplicates from a SELECT statement’s result set.

**Q72. What are the syntax and use of the COALESCE function?**

From a succession of expressions, the COALESCE function returns the first non-NULL value. The expressions are evaluated in the order that they are supplied, and the function’s result is the first non-null value. Only if all of the inputs are null does the COALESCE method return NULL.

The syntax of COALESCE function is COALESCE (exp1, exp2, …. expn)

### **Q73. What is the ACID property in a database?**

ACID stands for Atomicity, Consistency, Isolation, Durability. It is used to ensure that the data transactions are processed reliably in a database system.

* **Atomicity:** Atomicity refers to the transactions that are completely done or failed where transaction refers to a single logical operation of a data. It means if one part of any transaction fails, the entire transaction fails and the database state is left unchanged.
* **Consistency:** Consistency ensures that the data must meet all the validation rules. In simple words,  you can say that your transaction never leaves the database without completing its state.
* **Isolation:** The main goal of isolation is concurrency control.
* **Durability:** Durability means that if a transaction has been committed, it will occur whatever may come in between such as power loss, crash or any sort of error.

### **Q74. What do you mean by “Trigger” in SQL?**

[Trigger in SQL](https://www.edureka.co/blog/triggers-in-sql/) is are a special type of stored procedures that are defined to execute automatically in place or after data modifications. It allows you to execute a batch of code when an insert, update or any other query is executed against a specific table.

### **Q75. What are the different operators available in SQL?**

There are three [operators available in SQL](https://www.edureka.co/blog/sql-operators/), namely:

1. Arithmetic Operators
2. Logical Operators
3. Comparison Operators

Apart from this SQL Interview Questions blog, if you want to get trained from professionals on this technology, you can opt for structured training from edureka!

### **Q76.  Are NULL values same as that of zero or a blank space?**

A NULL value is not at all same as that of zero or a blank space. NULL value represents a value which is unavailable, unknown, assigned or not applicable whereas a zero is a number and blank space is a character.

### **Q77. What is the difference between cross join and natural join?**

The cross join produces the cross product or Cartesian product of two tables whereas the natural join is based on all the columns having the same name and data types in both the tables.

### **Q78. What is subquery in SQL?**

A subquery is a query inside another query where a query is defined to retrieve data or information back from the database. In a subquery, the outer query is called as the main query whereas the inner query is called subquery. Subqueries are always executed first and the result of the subquery is passed on to the main query. It can be nested inside a SELECT, UPDATE or any other query. A subquery can also use any comparison operators such as >,< or =.

### **Q79. What are the different types of a subquery?**

There are two types of subquery namely, Correlated and Non-Correlated.

**Correlated subquery**: These are queries which select the data from a table referenced in the outer query. It is not considered as an independent query as it refers to another table and refers the column in a table.

**Non-Correlated subquery**: This query is an independent query where the output of subquery is substituted in the main query.

Let’s move to the next question in this SQL Interview Questions.

### **Q80. List the ways to get the count of records in a table?**

To count the number of records in a [table in SQL](https://www.edureka.co/blog/create-table-in-sql/), you can use the below commands:

SELECT \* FROM table1

SELECT COUNT(\*) FROM table1

SELECT rows FROM sysindexes WHERE id = OBJECT\_ID(table1) AND indid < 2

Apart from this SQL Interview Questions Blog, if you want to get trained from professionals on this technology, you can opt for [structured training from edureka!](https://www.edureka.co/sql-essentials-training)

### **Q81. Write a SQL query to find the names of employees that begin with ‘A’?**

To display name of the employees that begin with ‘A’, type in the below command:

|  |  |
| --- | --- |
| 1 | SELECT \* FROM Table\_name WHERE EmpName like 'A%' |

### **Q82. Write a SQL query to get the third-highest salary of an employee from employee\_table?**

|  |  |
| --- | --- |
| 1  2  3  4  5  6 | SELECT TOP 1 salary  FROM(  SELECT TOP 3 salary  FROM employee\_table  ORDER BY salary DESC) AS emp  ORDER BY salary ASC; |

### **Q83. What is the need for group functions in SQL?**

Group functions work on the set of rows and return one result per group. Some of the commonly used group functions are: AVG, COUNT, MAX, MIN, SUM, VARIANCE.

### Q84. What is a Relationship and what are they?

Relation or links are between entities that have something to do with each other. Relationships are defined as the connection between the tables in a database. There are various relationships, namely:

* One to One Relationship.
* One to Many Relationship.
* Many to One Relationship.
* Self-Referencing Relationship.

### **Q85.  How can you insert NULL values in a column while inserting the data?**

NULL values in SQL can be inserted in the following ways:

* Implicitly by omitting column from column list.
* Explicitly by specifying NULL keyword in the VALUES clause

### **Q86. What is the main difference between ‘BETWEEN’ and ‘IN’ condition operators?**

BETWEEN operator is used to display rows based on a range of values in a row whereas the IN condition operator is used to check for values contained in a specific set of values.

### Example of BETWEEN:

SELECT \* FROM Students where ROLL\_NO BETWEEN 10 AND 50;

**Example of IN:**

SELECT \* FROM students where ROLL\_NO IN (8,15,25);

### Q87. Why are SQL functions used?

[SQL functions](https://www.edureka.co/blog/sql-functions) are used for the following purposes:

* To perform some calculations on the data
* To modify individual data items
* To manipulate the output
* To format dates and numbers
* To convert the data types

### **Q88. What is the need for MERGE statement?**

This statement allows conditional update or insertion of data into a table. It performs an UPDATE if a row exists, or an INSERT if the row does not exist.

### **Q89.** What do you mean by recursive stored procedure?

Recursive stored procedure refers to a stored procedure which calls by itself until it reaches some boundary condition. This recursive function or procedure helps the programmers to use the same set of code n number of times.

### **Q90. What is CLAUSE in SQL?**

SQL clause helps to limit the result set by providing a condition to the query. A clause helps to filter the rows from the entire set of records.

For example – WHERE, HAVING clause.

Apart from this SQL Interview Questions Blog, if you want to get trained from professionals on this technology, you can opt for a structured training from edureka! Click below to know more.

### **Q91. What is the difference between ‘HAVING’ CLAUSE and a ‘WHERE’ CLAUSE?**

HAVING clause can be used only with SELECT statement. It is usually used in a GROUP BY clause and whenever GROUP BY is not used, HAVING behaves like a WHERE clause.  
Having Clause is only used with the GROUP BY function in a query whereas WHERE Clause is applied to each row before they are a part of the GROUP BY function in a query.

### Q92. List the ways in which  Dynamic SQL can be executed?

Following are the ways in which dynamic SQL can be executed:

* Write a query with parameters.
* Using EXEC.
* Using sp\_executesql.

### Q93. What are the various levels of constraints?

Constraints are the representation of a column to enforce data entity and consistency. There are two levels  of a constraint, namely:

* column level constraint
* table level constraint

### **Q94. How can you fetch common records from two tables?**

You can fetch common records from two tables using INTERSECT. For example:

Select studentID from student. <strong>INTERSECT </strong> Select StudentID from Exam

### Q95. List some case manipulation functions in SQL?

There are three case manipulation functions in SQL, namely:

* LOWER: This function returns the string in lowercase. It takes a string as an argument and returns it by converting it into lower case. Syntax:

LOWER(‘string’)

* UPPER: This function returns the string in uppercase. It takes a string as an argument and returns it by converting it into uppercase. Syntax:

UPPER(‘string’)

* INITCAP: This function returns the string with the first letter in uppercase and rest of the letters in lowercase. Syntax:

INITCAP(‘string’)

Apart from this SQL Interview Questions blog, if you want to get trained from professionals on this technology, you can opt for a structured training from edureka! Click below to know more.

### Q96. What are the different set operators available in SQL?

Some of the available set operators are – Union, Intersect or Minus operators.

### Q97. What is an ALIAS command?

[ALIAS command in SQL](https://www.edureka.co/blog/sql-commands) is the name that can be given to any table or a column. This alias name can be referred in WHERE clause to identify a particular table or a column.

For example-

Select emp.empID, dept.Result from employee emp, department as dept where emp.empID=dept.empID

In the above example, emp refers to alias name for employee table and dept refers to alias name for department table.

Let’s move to the next question in this SQL Interview Questions.

### Q98. What are aggregate and scalar functions?

Aggregate functions are used to evaluate mathematical calculation and returns a single value. These calculations are done from the columns in a table. For example- max(),count() are calculated with respect to numeric.

Scalar functions return a single value based on the input value. For example – UCASE(), NOW() are calculated with respect to string.

Let’s move to the next question in this SQL Interview Questions.

### Q99. How can you fetch alternate records from a table?

You can fetch alternate records i.e both odd and even row numbers. For example- To display even numbers, use the following command:

Select studentId from (Select rowno, studentId from student) where mod(rowno,2)=0

Now, to display odd numbers:

Select studentId from (Select rowno, studentId from student) where mod(rowno,2)=1

### Q100. Name the operator which is used in the query for pattern matching?

LIKE operator is used for pattern matching, and it can be used as -.

1. % – It matches zero or more characters.

For example- select \* from students where studentname like ‘a%’

\_ (Underscore) – it matches exactly one character.  
For example- select \* from student where studentname like ‘abc\_’

Apart from this SQL Interview Questions Blog, if you want to get trained from professionals on this technology, you can opt for [structured training from edureka!](https://www.edureka.co/sql-essentials-training)

### Q101. How can you select unique records from a table?

You can select unique records from a table by using the DISTINCT keyword.

Select DISTINCT studentID from Student

Using this command, it will print unique student id from the table Student.

### Q102. How can you fetch first 5 characters of the string?

There are a lot of ways to fetch characters from a string. For example:

Select SUBSTRING(StudentName,1,5) as studentname from student

### **Q103.**What is the main difference between SQL and PL/SQL?

SQL is a query language that allows you to issue a single query or execute a single insert/update/delete whereas PL/SQL is Oracle’s “Procedural Language” SQL, which allows you to write a full program (loops, variables, etc.) to accomplish multiple operations such as selects/inserts/updates/deletes.

### **Q104. What is a View?**

A view is a virtual table which consists of a subset of data contained in a table. Since views are not present, it takes less space to store. View can have data of one or more tables combined and it depends on the relationship.

Let’s move to the next question in this SQL Interview Questions.

### **Q105. What are Views used for?**

A view refers to a logical snapshot based on a table or another view. It is used for the following reasons:

* Restricting access to data.
* Making complex queries simple.
* Ensuring data independence.
* Providing different views of same data.

### Q106. What is a Stored Procedure?

A Stored Procedure is a function which consists of many SQL statements to access the database system. Several SQL statements are consolidated into a stored procedure and execute them whenever and wherever required which saves time and avoid writing code again and again.

### Q107. List some advantages and disadvantages of Stored Procedure?

### **Advantages:**

A Stored Procedure can be used as a modular programming which means create once, store and call for several times whenever it is required. This supports faster execution. It also reduces network traffic and provides better security to the data.

### **Disadvantage:**

The only disadvantage of Stored Procedure is that it can be executed only in the database and utilizes more memory in the database server.

### Q108. List all the types of user-defined functions?

There are three types of user-defined functions, namely:

* Scalar Functions
* Inline Table-valued functions
* Multi-statement valued functions

Scalar returns the unit, variant defined the return clause. Other two types of defined functions return table.

Let’s move to the next question in this SQL Interview Questions.

### Q109. What do you mean by Collation?

Collation is defined as a set of rules that determine how data can be sorted as well as compared. Character data is sorted using the rules that define the correct character sequence along with options for specifying case-sensitivity, character width etc.

Let’s move to the next question in this SQL Interview Questions.

### Q110. What are the different types of Collation Sensitivity?

Following are the different types of collation sensitivity:

* Case Sensitivity: A and a and B and b.
* Kana Sensitivity: Japanese Kana characters.
* Width Sensitivity: Single byte character and double-byte character.
* Accent Sensitivity.

Apart from this SQL Interview Questions Blog, if you want to get trained from professionals on this technology, you can opt for[structured training from edureka!](https://www.edureka.co/sql-essentials-training)

### Q111. What are Local and Global variables?

### **Local variables:**

These variables can be used or exist only inside the function. These variables are not used or referred by any other function.

### **Global variables:**

These variables are the variables which can be accessed throughout the program. Global variables cannot be created whenever that function is called.

### Q112. What is Auto Increment in SQL?

Autoincrement keyword allows the user to create a unique number to get generated whenever a new record is inserted into the table.  
This keyword is usually required whenever PRIMARY KEY in SQL is used.

[AUTO INCREMENT keyword](https://www.edureka.co/blog/sql-auto-increment/) can be used in Oracle and IDENTITY keyword can be used in SQL SERVER.

### Q113. What is a Datawarehouse?

Datawarehouse refers to a central repository of data where the data is assembled from multiple sources of information. Those data are consolidated, transformed and made available for the mining as well as online processing. Warehouse data also have a subset of data called Data Marts.

### Q114. What are the different authentication modes in SQL Server? How can it be changed?

Windows mode and Mixed Mode – SQL and Windows. You can go to the below steps to change authentication mode in SQL Server:

* Click Start> Programs> Microsoft SQL Server and click SQL Enterprise Manager to run SQL Enterprise Manager from the Microsoft SQL Server program group.
* Then select the server from the Tools menu.
* Select SQL Server Configuration Properties, and choose the Security page.

### **Q115. What are STUFF and REPLACE function?**

**STUFF Function**: This function is used to overwrite existing character or inserts a string into another string. Syntax:

STUFF(string\_expression,start, length, replacement\_characters)

where,  
string\_expression: it is the string that will have characters substituted

start: This refers to the starting position  
length: It refers to the number of characters in the string which are substituted.

replacement\_string: They are the new characters which are injected in the string.

**REPLACE function**: This function is used to replace the existing characters of all the occurrences. Syntax:

REPLACE (string\_expression, search\_string, replacement\_string)

Here every search\_string in the string\_expression will be replaced with the replace

# ****R****

## 1. How to determine data type of an object?

**class()** is used to determine data type of an object. See the example below -

x <- factor(1:5)  
**class(x)**

It returns factor.

|  |
| --- |
| <https://4.bp.blogspot.com/-EALDouFnsEI/V8rVhExlJlI/AAAAAAAAFSs/a9kcxojmde8EHMK0ptpB-EX168_tAwLIgCLcB/s1600/class.png> |
| Object Class |

To determine structure of an object, use **str() function** :

str(x) returns "Factor w/ 5 level"

**Example 2 :**

xx <- data.frame(var1=c(1:5))  
**class(xx)**

It returns "data.frame".

**str(xx)** returns 'data.frame' : 5 obs. of  1 variable: $ var1: int

## 2.  What is the use of mode() function?

It returns the storage mode of an object.

x <- factor(1:5)  
**mode(x)**

The above mode function returns **numeric**.

|  |
| --- |
| <https://2.bp.blogspot.com/-yhHWvXtgJA4/V8raYfvr0OI/AAAAAAAAFS8/JFCafxvUpGQgLJUnkK90oYMN47DxLOmtACLcB/s1600/mode.png> |
| Mode Function |

x <- data.frame(var1=c(1:5))  
mode(x)

It returns list.

## 3. Which data structure is used to store categorical variables?

R has a special data structure called **"factor"** to store categorical variables. It tells R that a variable is nominal or ordinal by making it a factor.

gender = c(1,2,1,2,1,2)  
gender = **factor**(gender)  
gender

## 4. How to check the frequency distribution of a categorical variable?

The **table function** is used to calculate the count of each categories of a categorical variable.

gender = factor(c("m","f","f","m","f","f"))  
table(gender)

|  |
| --- |
| <https://4.bp.blogspot.com/-GZHWjsrc1Fk/V8rbTNDk_NI/AAAAAAAAFTA/0TfkFoVBSE8_K1PvljilNaggWuJUYW5WQCLcB/s1600/table.png> |
| Output |

If you want to include **% of values in each group**, you can store the result in data frame using data.frame function and the calculate the column percent.

t = data.frame(table(gender))  
t$percent= round(t$Freq / sum(t$Freq)\*100,2)

|  |
| --- |
| <https://4.bp.blogspot.com/-1f3hMs606Co/V8rd4I7b6PI/AAAAAAAAFTM/FXNWS4FcYQUdPCZ_jD0-gOrAk5k1KKRPACLcB/s1600/table2.png> |
| Frequency Distribution |

## How to check the cumulative frequency distribution of a categorical variable

The **cumsum** function is used to calculate the cumulative sum of a categorical variable.

gender = factor(c("m","f","f","m","f","f"))  
x = **table**(gender)  
**cumsum**(x)

|  |
| --- |
| <https://1.bp.blogspot.com/-wnwHcyBuWrU/V8rfT1SV8mI/AAAAAAAAFTY/II34ngYJAa4g-cAB_VaeCfRby92NX-fsACLcB/s1600/cumsum.png> |
| Cumulative Sum |

If you want to see the **cumulative percentage of values**, see the code below :

t = data.frame(table(gender))  
t$cumfreq = cumsum(t$Freq)  
t$cumpercent= round(t$cumfreq / sum(t$Freq)\*100,2)

|  |
| --- |
| <https://2.bp.blogspot.com/-ezNM4mx2YQY/V8rgJPq01NI/AAAAAAAAFTc/9HGINdCB_jwHet-6069MMGygbt2Vh8zQACLcB/s1600/cumsum2.png> |
| Cumulative Frequency Distribution |

## 6. How to produce histogram

The **hist** function is used to produce the histogram of a variable.

df = sample(1:100, 25)  
hist(df, right=FALSE)

|  |
| --- |
| <https://4.bp.blogspot.com/-UmzI4tGWCjw/V1gpEylnnGI/AAAAAAAAEfk/Te8qDUPnxaQzif21IT-o-fU8Gb61xrv5QCLcB/s1600/Rplot01.png> |
| Produce Histogram with R |

***To improve the layout of histogram, you can use the code below***

colors = c("red", "yellow", "green", "violet", "orange", "blue", "pink", "cyan")  
hist(df,  right=FALSE,  col=colors, main="Main Title ", xlab="X-Axis Title")

## 7. How to produce bar graph

First calculate the frequency distribution with **table** function and then apply **barplot** function to produce bar graph

mydata = sample(LETTERS[1:5],16,replace = TRUE)  
mydata.count= table(mydata)  
barplot(mydata.count)

***To improve the layout of bar graph, you can use the code below:***

colors = c("red", "yellow", "green", "violet", "orange", "blue", "pink", "cyan")  
barplot(mydata.count, col=colors, main="Main Title ", xlab="X-Axis Title")

|  |
| --- |
| <https://1.bp.blogspot.com/-ZWt4qcWrSpk/V1guMRNx1uI/AAAAAAAAEf0/Ko_2uhuUKyQ74B-tYanUmVkmQx0C9amVgCLcB/s1600/Rplot02.png> |
| Bar Graph with R |

## 8. How to produce Pie Chart

First calculate the frequency distribution with **table** function and then apply **pie** function to produce pie chart.

mydata = sample(LETTERS[1:5],16,replace = TRUE)  
mydata.count= table(mydata)  
pie(mydata.count, col=rainbow(12))

|  |
| --- |
| <https://2.bp.blogspot.com/-p57trsNs6-0/V8rg63qTBwI/AAAAAAAAFTk/mRQukw9ZBDcnfCwr0X56yVr6Xkf5QwCFQCLcB/s1600/pie%2Bchart.png> |
| Pie Chart with R |

## 9. Multiplication of 2 vectors having different length

For example, you have two vectors as defined below -

x <- c(4,5,6)  
y <- c(2,3)

**If you run this vector z <- x\*y , what would be the output? What would be the length of z?**

It returns 8 15 12 with the warning message as shown below. The length of z is 3 as it has three elements.

|  |
| --- |
| <https://4.bp.blogspot.com/-1gLu3L-iQfk/V8rjjxcdpuI/AAAAAAAAFTs/Th4LklWxGX48FV4h1n6fqhZa--oCUJVWQCLcB/s1600/multiply.png> |
| Multiplication of vectors |

**First Step :** It performs multiplication of the first element of vector x i.e. 4 with first element of vector y i.e. 2 and the result is 8. In the **second step,** it multiplies second element of vector x i.e. 5 with second element of vector b i.e. 3, and the result is 15. In the next step, R multiplies first element of smaller vector (y) with last element of bigger vector x.

Suppose the vector x would contain four elements as shown below :

x <- c(4,5,6,7)  
y <- c(2,3)  
x\*y

It returns 8 15 12 21. It works like this : (4\*2) (5\*3) (6\*2) (7\*3)

## (!!!) What are the different data structures R contain?

R contains primarily the following data structures :

1. **Vector**
2. **Matrix**
3. **Array**
4. **List**
5. **Data frame**
6. **Factor**

The first three data types (vector, matrix, array) are **homogeneous** in behavior. It means all contents must be of the same type. The fourth and fifth data types (list, data frame) are **heterogeneous** in behavior. It implies they allow different types. And the factor data type is used to store categorical variable.

**Data Types**  
  
Unlike SAS and SPSS, R has several different data types (structures) including vectors, factors, data frames, matrices, arrays, and lists. The data frame is most like a dataset in SAS.

### 1. Vectors

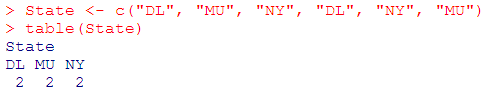
A vector is an object that contains a set of values called its elements.  
  
**Numeric vector**

x <- c(1,2,3,4,5,6)

*The operator <–  is equivalent to "=" sign.*  
  
**Character vector**

State <- c("DL", "MU", "NY", "DL", "NY", "MU")

*To calculate frequency for State vector, you can use* ***table*** *function.*

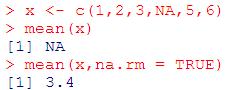
[](https://3.bp.blogspot.com/-SAq5Eyx51yU/U51DZ-OBGYI/AAAAAAAADBY/RAvt1qcdbuY/s1600/table.png)

*To calculate mean for a vector, you can use* ***mean*** *function.*

[https://2.bp.blogspot.com/--qnDR9o4dCE/U51HPWE7a1I/AAAAAAAADBw/oqy81JEBwNU/s1600/mean.png](https://2.bp.blogspot.com/--qnDR9o4dCE/U51HPWE7a1I/AAAAAAAADBw/oqy81JEBwNU/s1600/mean.png)

*Since the above vector contains a NA (not available) value, the mean function returns NA.*

*To calculate mean for a vector* ***excluding NA values****, you can include****na.rm = TRUE*** *parameter in mean function.*

[](https://1.bp.blogspot.com/-PlyeQsrKQE4/U51IImZGpvI/AAAAAAAADB4/R0igYeHDNzY/s1600/mean.png)

You can use subscripts to refer elements of a vector.

[https://1.bp.blogspot.com/-AiprM-QjENU/U51FNihawyI/AAAAAAAADBk/I9REsgyyX3Y/s1600/element.png](https://1.bp.blogspot.com/-AiprM-QjENU/U51FNihawyI/AAAAAAAADBk/I9REsgyyX3Y/s1600/element.png)

**Convert a column "x" to numeric**

data$x = as.numeric(data$x)

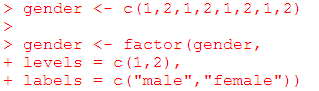
### 2. Factors

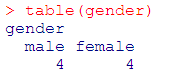
R has a special data structure to store ***categorical variables***. It tells R that a variable is nominal or ordinal by making it a factor.

Simplest form of the factor function :

[https://2.bp.blogspot.com/-e6A3R0Xb5vU/U51apbhsrTI/AAAAAAAADCo/P9xIbHv0x0g/s1600/simplest.png](https://2.bp.blogspot.com/-e6A3R0Xb5vU/U51apbhsrTI/AAAAAAAADCo/P9xIbHv0x0g/s1600/simplest.png)

Ideal form of the factor function :

[](https://3.bp.blogspot.com/-Ykcf7OADIS0/U51XIVVDI8I/AAAAAAAADCQ/iThgiIt5Z5k/s1600/gendere.png)

[](https://2.bp.blogspot.com/-dcxYO_rgPVc/U51YkR1GUYI/AAAAAAAADCc/lyxhC0o4GRY/s1600/gendere.png)

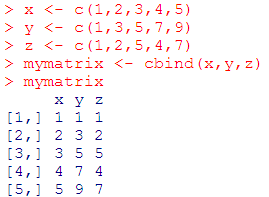
The factor function has three parameters:

1. Vector Name
2. Values (Optional)
3. Value labels (Optional)

**Convert a column "x" to factor**

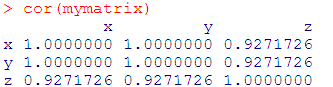
data$x = as.factor(data$x)

### 3. Matrices

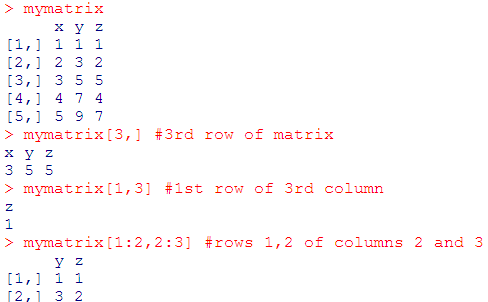
All values in columns in a matrix must have the same mode (numeric, character, etc.) and the same length.  
  
The **cbind** function joins columns together into a matrix. See the usage below  
  
[](https://4.bp.blogspot.com/-UiN5SxeduMA/U51fG9-CnwI/AAAAAAAADC0/eJHYt8WwKjk/s1600/matrix.png)  
  
The numbers to the left side in brackets are the row numbers. The form [1, ] means that it is row number one and the blank following the comma means that R has displayed all the columns.  
  
To see dimension of the matrix, you can use **dim** function.

[https://3.bp.blogspot.com/-1jD-DKUN4R0/U51gcv4OLjI/AAAAAAAADDA/qMZR_z7Dn7Q/s1600/dimesnsion.png](https://3.bp.blogspot.com/-1jD-DKUN4R0/U51gcv4OLjI/AAAAAAAADDA/qMZR_z7Dn7Q/s1600/dimesnsion.png)

To see correlation of the matrix, you can use **cor** function.

[](https://4.bp.blogspot.com/-YZkdQWpgohU/U51hN__tTYI/AAAAAAAADDI/6eqZMTpCk7A/s1600/correlation.png)

You can use subscripts to identify rows or columns.

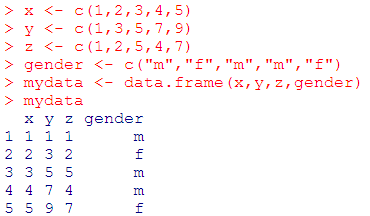
[](https://2.bp.blogspot.com/-NO-u9RTJ3hU/U51jTTeissI/AAAAAAAADDU/KueeqokMmR8/s1600/rows+and+columns.png)

### 4. Arrays

Arrays are similar to matrices but can have more than two dimensions.

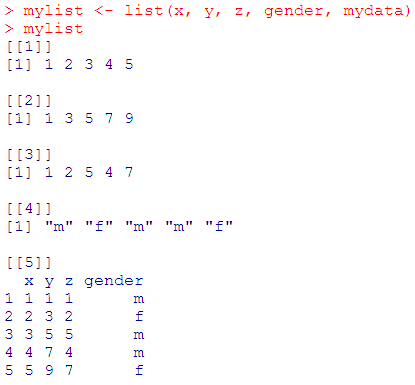
### 5. Data Frames

A data frame is similar to SAS and SPSS datasets. It contains variables and records.  
It is more general than a matrix, in that different columns can have different modes (numeric, character, factor, etc.  
  
The **data.frame** function is used to combine variables (vectors and factors) into a data frame.

[](https://1.bp.blogspot.com/-ARCRjI-YC9E/U51mMJvCjoI/AAAAAAAADDg/0JjHsifRK3Y/s1600/data+frame1.png)

### 6. Lists

A list allows you to store a variety of  objects.

[](https://1.bp.blogspot.com/-KLZBqwsbMYo/U51nvO0E8EI/AAAAAAAADDs/RN1nFnU3tOE/s1600/sasas.png)

You can use subscripts to select the specific component of the list.

[https://3.bp.blogspot.com/-HghkPlsfkvw/U51oz0yC2KI/AAAAAAAADD0/Ttd_iNnT24Y/s1600/subsc.png](https://3.bp.blogspot.com/-HghkPlsfkvw/U51oz0yC2KI/AAAAAAAADD0/Ttd_iNnT24Y/s1600/subsc.png)

### How to know data type of a column

1. **'class'** is a property assigned to an object that determines how generic functions operate with it.  It is not a mutually exclusive classification.  
  
2. **'mode'** is a mutually exclusive classification of objects according to their basic structure.  The 'atomic' modes are numeric, complex, charcter and logical.  
  
> x <- 1:16  
> x <- factor(x)  
  
> class(x)  
[1] "factor"  
  
> mode(x)  
[1] "numeric"

## 11. How to combine data frames?

Let's prepare 2 vectors for demonstration :

x = c(1:5)  
y = c("m","f","f","m","f")

The **cbind() function** is used to combine data frame by **columns**.

z=cbind(x,y)

|  |
| --- |
| <https://3.bp.blogspot.com/-1b73-Jk9Whs/V8r0eqbaZnI/AAAAAAAAFT8/slVmmLkddhAvPNY-vsv6HwekgpekF4DFACLcB/s1600/combine.png> |
| cbind : Output |

The **rbind() function** is used to combine data frame by **rows**.

z = rbind(x,y)

|  |
| --- |
| <https://3.bp.blogspot.com/-VQctPEUcAKU/V8r2BNiUh6I/AAAAAAAAFUE/WMJmXyL4Ra0RCsdJueO4RnrpKNe6GVKwwCLcB/s1600/rbind.png> |
| rbind : Output |

While using **cbind()** function, make sure the **number of rows must be equal** in both the datasets. While using **rbind()** function, make sure both the **number and names of columns** must be same. If names of columns would not be same, wrong data would be appended to columns or records might go missing.

## 12. How to combine data by rows when different number of columns?

When the number of columns in datasets are not equal, **rbind() function doesn't work** to combine data by rows. For example, we have two data frames df and df2. The data frame df has 2 columns and df2 has only 1 variable. See the code below -

df = data.frame(x = c(1:4), y = c("m","f","f","m"))  
df2 = data.frame(x = c(5:8))

The **bind\_rows()** function from dplyr package can be used to combine data frames when number of columns do not match.

library(dplyr)  
combdf = **bind\_rows**(df,df2)

## Important dplyr Functions to remember

|  |  |  |
| --- | --- | --- |
| **dplyr Function** | **Description** | **Equivalent SQL** |
| select() | Selecting columns (variables) | SELECT |
| filter() | Filter (subset) rows. | WHERE |
| group\_by() | Group the data | GROUP BY |
| summarise() | Summarise (or aggregate) data | - |
| arrange() | Sort the data | ORDER BY |
| join() | Joining data frames (tables) | JOIN |
| mutate() | Creating New Variables | COLUMN ALIAS |

### **dplyr vs. Base R Functions**

dplyr functions process faster than base R functions. It is because dplyr functions were written in a computationally efficient manner. They are also more stable in the syntax and better supports data frames than vectors.

SQL Queries vs. dplyr

People have been utilizing SQL for analyzing data for decades. Every modern data analysis software such as Python, R, SAS etc supports SQL commands. But SQL was never designed to perform data analysis. It was rather designed for querying and managing data. There are many data analysis operations where SQL fails or makes simple things difficult. For example, calculating median for multiple variables, converting wide format data to long format etc. Whereas, dplyr package was designed to do data analysis.

The names of dplyr functions are similar to SQL commands such as select() for selecting variables, group\_by() - group data by grouping variable, join() - joining two data sets. Also includes inner\_join() and left\_join(). It also supports sub queries for which SQL was popular for.

Data : Income Data by States

In this tutorial, we are using the following data which contains income generated by states from year 2002 to 2015. **Note :** This data do not contain actual income figures of the states. To download the dataset, click on this link - [**Dataset**](https://raw.githubusercontent.com/deepanshu88/data/master/sampledata.csv) and then right click and hit Save as option.   
This dataset contains 51 observations (rows) and 16 variables (columns). The snapshot of first 6 rows of the dataset is shown below.

Index State Y2002 Y2003 Y2004 Y2005 Y2006 Y2007 Y2008 Y2009

1 A Alabama 1296530 1317711 1118631 1492583 1107408 1440134 1945229 1944173

2 A Alaska 1170302 1960378 1818085 1447852 1861639 1465841 1551826 1436541

3 A Arizona 1742027 1968140 1377583 1782199 1102568 1109382 1752886 1554330

4 A Arkansas 1485531 1994927 1119299 1947979 1669191 1801213 1188104 1628980

5 C California 1685349 1675807 1889570 1480280 1735069 1812546 1487315 1663809

6 C Colorado 1343824 1878473 1886149 1236697 1871471 1814218 1875146 1752387

Y2010 Y2011 Y2012 Y2013 Y2014 Y2015

1 1237582 1440756 1186741 1852841 1558906 1916661

2 1629616 1230866 1512804 1985302 1580394 1979143

3 1300521 1130709 1907284 1363279 1525866 1647724

4 1669295 1928238 1216675 1591896 1360959 1329341

5 1624509 1639670 1921845 1156536 1388461 1644607

6 1913275 1665877 1491604 1178355 1383978 1330736

How to load Data

Submit the following code to load data directly from link. If you want to load the data from your local drive, you need to change the file path in the code below.

mydata = read.csv("https://raw.githubusercontent.com/deepanshu88/data/master/sampledata.csv")

dplyr Practical Examples

Example 1 : Selecting Random N Rows

The **sample\_n** function selects random rows from a data frame (or table). The second parameter of the function tells R the number of rows to select.

sample\_n(mydata,3)

Index State Y2002 Y2003 Y2004 Y2005 Y2006 Y2007 Y2008 Y2009

2 A Alaska 1170302 1960378 1818085 1447852 1861639 1465841 1551826 1436541

8 D Delaware 1330403 1268673 1706751 1403759 1441351 1300836 1762096 1553585

33 N New York 1395149 1611371 1170675 1446810 1426941 1463171 1732098 1426216

Y2010 Y2011 Y2012 Y2013 Y2014 Y2015

2 1629616 1230866 1512804 1985302 1580394 1979143

8 1370984 1318669 1984027 1671279 1803169 1627508

33 1604531 1683687 1500089 1718837 1619033 1367705

Example 2 : Selecting Random Fraction of Rows

The **sample\_frac** function returns randomly N% of rows. In the example below, it returns randomly 10% of rows.

sample\_frac(mydata,0.1)

Example 3 : Remove Duplicate Rows based on all the variables (Complete Row)

The **distinct function** is used to eliminate duplicates.

x1 = distinct(mydata)

In this dataset, there is not a single duplicate row so it returned same number of rows as in mydata.

Example 4 : Remove Duplicate Rows based on a variable

The **.keep\_all** function is used to retain all other variables in the output data frame.

x2 = distinct(mydata, Index, .keep\_all= TRUE)

Example 5 : Remove Duplicates Rows based on multiple variables

In the example below, we are using two variables - **Index, Y2010** to determine uniqueness.

x2 = distinct(mydata, Index, Y2010, .keep\_all= TRUE)

### **select( ) Function**

It is used to select only desired variables.

select() syntax : select(data , ....)

data : Data Frame

.... : Variables by name or by function

Example 6 : Selecting Variables (or Columns)

Suppose you are asked to select only a few variables. The code below selects variables "Index", columns from "State" to "Y2008".

mydata2 = select(mydata, Index, State:Y2008)

Example 7 : Dropping Variables

The **minus sign** before a variable tells R to drop the variable.

mydata = select(mydata, -Index, -State)

The above code can also be written like :

mydata = select(mydata, -c(Index,State))

Example 8 : Selecting or Dropping Variables starts with 'Y'

The **starts\_with()** function is used to select variables starts with an alphabet.

mydata3 = select(mydata, starts\_with("Y"))

Adding a negative sign before starts\_with() implies dropping the variables starts with 'Y'

mydata33 = select(mydata, -starts\_with("Y"))

***The following functions helps you to select variables based on their names.***

|  |  |
| --- | --- |
| **Helpers** | **Description** |
| starts\_with() | Starts with a prefix |
| ends\_with() | Ends with a prefix |
| contains() | Contains a literal string |
| matches() | Matches a regular expression |
| num\_range() | Numerical range like x01, x02, x03. |
| one\_of() | Variables in character vector. |
| everything() | All variables. |

Example 9 : Selecting Variables contain 'I' in their names

mydata4 = select(mydata, contains("I"))

Example 10 : Reorder Variables

The code below keeps variable **'State'** in the front and the remaining variables follow that.

mydata5 = select(mydata, State, everything())

New order of variables are displayed below -

[1] "State" "Index" "Y2002" "Y2003" "Y2004" "Y2005" "Y2006" "Y2007" "Y2008" "Y2009"

[11] "Y2010" "Y2011" "Y2012" "Y2013" "Y2014" "Y2015"

### **rename( ) Function**

It is used to change variable name.

rename() syntax : rename(data , new\_name = old\_name)

data : Data Frame

new\_name : New variable name you want to keep

old\_name : Existing Variable Name

Example 11 : Rename Variables

The rename function can be used to rename variables.  
  
In the following code, we are renaming **'Index'** variable to **'Index1'**.

mydata6 = rename(mydata, Index1=Index)

|  |
| --- |
| <https://1.bp.blogspot.com/-Uvip9fEk4BA/V7GuKZ-7cwI/AAAAAAAAFDk/-6btohOqZOoCrKjygYOFReMX5orfaFVDQCLcB/s1600/rename.png> |
| Output |

### **filter( ) Function**

It is used to subset data with matching logical conditions.

filter() syntax : filter(data , ....)

data : Data Frame

.... : Logical Condition

Example 12 : Filter Rows

Suppose you need to subset data. You want to filter rows and retain only those values in which Index is equal to A.

mydata7 = filter(mydata, Index == "A")

Index State Y2002 Y2003 Y2004 Y2005 Y2006 Y2007 Y2008 Y2009

1 A Alabama 1296530 1317711 1118631 1492583 1107408 1440134 1945229 1944173

2 A Alaska 1170302 1960378 1818085 1447852 1861639 1465841 1551826 1436541

3 A Arizona 1742027 1968140 1377583 1782199 1102568 1109382 1752886 1554330

4 A Arkansas 1485531 1994927 1119299 1947979 1669191 1801213 1188104 1628980

Y2010 Y2011 Y2012 Y2013 Y2014 Y2015

1 1237582 1440756 1186741 1852841 1558906 1916661

2 1629616 1230866 1512804 1985302 1580394 1979143

3 1300521 1130709 1907284 1363279 1525866 1647724

4 1669295 1928238 1216675 1591896 1360959 1329341

Example 13 : Multiple Selection Criteria

The **%in%** operator can be used to select multiple items. In the following program, we are telling R to select rows against 'A' and 'C' in column 'Index'.

mydata7 = filter(mydata6, Index %in% c("A", "C"))

Example 14 : 'AND' Condition in Selection Criteria

Suppose you need to apply 'AND' condition. In this case, we are picking data for 'A' and 'C' in the column 'Index' and income greater than 1.3 million in Year 2002.

mydata8 = filter(mydata6, Index %in% c("A", "C") & Y2002 >= 1300000 )

Example 15 : 'OR' Condition in Selection Criteria

The 'I' denotes OR in the logical condition. It means any of the two conditions.

mydata9 = filter(mydata6, Index %in% c("A", "C") | Y2002 >= 1300000)

Example 16 : NOT Condition

The "!" sign is used to reverse the logical condition.

mydata10 = filter(mydata6, !Index %in% c("A", "C"))

Example 17 : CONTAINS Condition

The **grepl function** is used to search for pattern matching. In the following code, we are looking for records wherein column **state** contains **'Ar'** in their name.

mydata10 = filter(mydata6, grepl("Ar", State))

### **summarise( ) Function**

It is used to summarize data.

summarise() syntax : summarise(data , ....)

data : Data Frame

..... : Summary Functions such as mean, median etc

Example 18 : Summarize selected variables

In the example below, we are calculating mean and median for the variable Y2015.

summarise(mydata, Y2015\_mean = mean(Y2015), Y2015\_med=median(Y2015))

|  |
| --- |
| <https://2.bp.blogspot.com/-v5xZFXf2iic/V7H7qgxFBnI/AAAAAAAAFD0/wmnDRIuMb2QSUafRfv7Er0BhhE0tr0vqwCLcB/s1600/mean_median.png> |
| Output |

Example 19 : Summarize Multiple Variables

In the following example, we are calculating number of records, mean and median for variables Y2005 and Y2006. The **summarise\_at** function allows us to select multiple variables by their names.

summarise\_at(mydata, vars(Y2005, Y2006), funs(n(), mean, median))

funs( ) has been soft-deprecated (dropped) from dplyr 0.8.0. Instead we should use list. The equivalent code is stated below -

summarise\_at(mydata, vars(Y2005, Y2006), list(n=~n(), mean=mean, median=median))

Another way of using it without stating names is through formula instead of function. This is mean = mean function and this is ~mean(.) formula.

summarise\_at(mydata, vars(Y2005, Y2006), list(~n(), ~mean(.), ~median(.)))

You must be wondering about ~ and . symbols. It's a way to pass purrr style anonymous function. See the base R method as compared to purrr style below. Both returns the same output. purrr style provides a shortcut to define anonymous function.

**Base R Style**

summarise\_at(mydata, vars(Y2005, Y2006), function(x) length(unique(x)))

**purrr Style**

summarise\_at(mydata, vars(Y2005, Y2006), ~length(unique(.)))

Symbols . and .x means the same thing. You can try the above code by replacing . with .x

|  |
| --- |
| <https://1.bp.blogspot.com/-WmYQ1a-zv_k/V7H9X6PrcVI/AAAAAAAAFEA/5Aup54UhokcV9ot9_v5QfqQ6IXdlpl4EwCLcB/s1600/multiplevars.png> |
| Output |

Example 20 : Summarize with Custom Functions

Incase you want to add additional arguments for the functions mean and median (for example **na.rm = TRUE**), you can do it like the code below.

summarise\_at(mydata, vars(Y2011, Y2012),funs(mean, median), na.rm = TRUE)

We can also use custom functions in the summarise function. In this case, we are computing the number of records, number of missing values, mean and median for variables Y2011 and Y2012. The **dot (.)** denotes each variables specified in the second argument of the function.

summarise\_at(mydata, vars(Y2011, Y2012),

funs(n(), missing = sum(is.na(.)), mean(., na.rm = TRUE), median(.,na.rm = TRUE)))

Instead of funs( ), you should make a habit of using list( ) as funs( ) can be dropped in future versions of dplyr package.

summarise\_at(mydata, vars(Y2011, Y2012),

list(~n(), missing = ~sum(is.na(.)), ~mean(., na.rm = TRUE),

~median(.,na.rm = TRUE)))

|  |
| --- |
| <https://3.bp.blogspot.com/-vIpoB0ujhAU/V7IA7OReoII/AAAAAAAAFEM/2dyXunS8BXYO86sTuXFjcDua9YwD-fexQCLcB/s1600/summarise.png> |
| Summarize : Output |

How to apply Non-Standard Functions

Suppose you want to subtract mean from its original value and then calculate variance of it.

set.seed(222)

mydata <- data.frame(X1=sample(1:100,100), X2=runif(100))

summarise\_at(mydata,vars(X1,X2), function(x) var(x - mean(x)))

X1 X2

1 841.6667 0.08142161

Equivalent purrr style method can be written like this :

summarise\_at(mydata,vars(X1,X2), ~ var(. - mean(.)))

Example 21 : Summarize all Numeric Variables

The **summarise\_if** function allows you to summarise conditionally.

summarise\_if(mydata, is.numeric, funs(n(),mean,median))

Alternative Method :

**First,** store data for all the numeric variables

numdata = mydata[sapply(mydata,is.numeric)]

**Second,** the **summarise\_all** function calculates summary statistics for all the columns in a data frame

summarise\_all(numdata, funs(n(),mean,median))

Example 22 : Summarize Factor Variable

We are checking the **number of levels/categories** and **count of missing observations** in a categorical (factor) variable.

summarise\_all(mydata["Index"], funs(nlevels(.), nmiss=sum(is.na(.))))

nlevels nmiss

1 19 0

### **arrange() function**

**Use :** Sort data

Syntax

arrange(data\_frame, variable(s)\_to\_sort)

or

data\_frame %>% arrange(variable(s)\_to\_sort)

To sort a variable in descending order, use **desc(x)**.

Example 23 : Sort Data by Multiple Variables

The default sorting order of **arrange() function** is ascending. In this example, we are sorting data by multiple variables.

arrange(mydata, Index, Y2011)

Suppose you need to sort one variable by descending order and other variable by ascending oder.

arrange(mydata, desc(Index), Y2011)

### **Pipe Operator %>%**

It is important to understand the pipe (%>%) operator before knowing the other functions of dplyr package. dplyr utilizes pipe operator from another package **(magrittr)**.

It allows you to write sub-queries like we do it in sql.

**Note :** All the functions in dplyr package can be used **without** the pipe operator. The question arises **"Why to use pipe operator %>%". The answer is** it lets to wrap multiple functions together with the use of %>%.

Syntax :

filter(data\_frame, variable == value)

or

data\_frame %>% filter(variable == value)

**The %>% is NOT restricted to filter function. It can be used with any function.**

Example :

The code below demonstrates the usage of pipe %>% operator. In this example, we are selecting 10 random observations of two variables "Index" "State" from the data frame "mydata".

dt = sample\_n(select(mydata, Index, State),10)

or

dt = mydata %>% select(Index, State) %>% sample\_n(10)

|  |
| --- |
| <https://4.bp.blogspot.com/-sVXtFxsUS88/V7h1xuJr5TI/AAAAAAAAFFs/AVx2PRLLl9oXZiycc7ZhjhC5-cAdtEDAgCLcB/s1600/pipe%2Boperator.png> |
| Output |

### **group\_by() function**

**Use :** Group data by categorical variable

Syntax :

group\_by(data, variables)

or

data %>% group\_by(variables)

Example 24 : Summarise Data by Categorical Variable

We are calculating count and mean of variables Y2011 and Y2012 by variable Index.

t = summarise\_at(group\_by(mydata, Index), vars(Y2011, Y2012), funs(n(), mean(., na.rm = TRUE)))

The above code can also be written like

t = mydata %>% group\_by(Index) %>%

summarise\_at(vars(Y2011:Y2015), funs(n(), mean(., na.rm = TRUE)))

Index Y2011\_n Y2012\_n Y2013\_n Y2014\_n Y2015\_n Y2011\_mean Y2012\_mean

A 4 4 4 4 4 1432642 1455876

C 3 3 3 3 3 1750357 1547326

D 2 2 2 2 2 1336059 1981868

F 1 1 1 1 1 1497051 1131928

G 1 1 1 1 1 1851245 1850111

H 1 1 1 1 1 1902816 1695126

I 4 4 4 4 4 1690171 1687056

K 2 2 2 2 2 1489353 1899773

L 1 1 1 1 1 1210385 1234234

M 8 8 8 8 8 1582714 1586091

N 8 8 8 8 8 1448351 1470316

O 3 3 3 3 3 1882111 1602463

P 1 1 1 1 1 1483292 1290329

R 1 1 1 1 1 1781016 1909119

S 2 2 2 2 2 1381724 1671744

T 2 2 2 2 2 1724080 1865787

U 1 1 1 1 1 1288285 1108281

V 2 2 2 2 2 1482143 1488651

W 4 4 4 4 4 1711341 1660192

Since dplyr >= 1.0.0 version you may get the following warnings.

#`summarise()` ungrouping output (override with `.groups` argument)

#`summarise()` regrouping output by xxx (override with `.groups` argument)

To suppress this warning you can use the following command.

options(dplyr.summarise.inform=F)

### **do() function**

**Use :** Compute within groups

Syntax :

do(data\_frame, expressions\_to\_apply\_to\_each\_group)

**Note :** *The* ***dot (.)*** *is required to refer to a data frame.*

Example 25 : Filter Data within a Categorical Variable

Suppose you need to pull top 2 rows from 'A', 'C' and 'I' categories of variable Index.

t = mydata %>% filter(Index %in% c("A", "C","I")) %>% group\_by(Index) %>%

do(head( . , 2))

|  |
| --- |
| <https://4.bp.blogspot.com/-wWWt_FBv-1Y/V7iIzjPLD8I/AAAAAAAAFF8/DN-txkfaGYY9xbUfs2E2V_7OrXo3EaL_ACLcB/s1600/do%2Boperation.png> |
| Output : do() function |

Example 26 : Selecting 3rd Maximum Value by Categorical Variable

We are calculating third maximum value of variable Y2015 by variable Index. The following code first selects only two variables Index and Y2015. Then it filters the variable Index with 'A', 'C' and 'I' and then it groups the same variable and sorts the variable Y2015 in descending order. At last, it selects the third row.

t = mydata %>% select(Index, Y2015) %>%

filter(Index %in% c("A", "C","I")) %>%

group\_by(Index) %>%

do(arrange(.,desc(Y2015))) %>% slice(3)

The **slice() function** is used to select rows by position.

|  |
| --- |
| <https://4.bp.blogspot.com/-mITEYVNCgHQ/V7iQe3MbkhI/AAAAAAAAFGM/z9tba5nf5ZIzGKaiOSipJJDpee8_ehzJQCLcB/s1600/rank%2Bfunction.png> |
| Output |

Using Window Functions

Like SQL, dplyr uses window functions that are used to subset data within a group. It returns a vector of values. We could use **min\_rank() function** that calculates rank in the preceding example,

t = mydata %>% select(Index, Y2015) %>%

filter(Index %in% c("A", "C","I")) %>%

group\_by(Index) %>%

filter(min\_rank(desc(Y2015)) == 3)

Index Y2015

1 A 1647724

2 C 1330736

3 I 1583516

Example 27 : Summarize, Group and Sort Together

In this case, we are computing mean of variables Y2014 and Y2015 by variable Index. Then sort the result by calculated mean variable Y2015.

t = mydata %>%

group\_by(Index)%>%

summarise(Mean\_2014 = mean(Y2014, na.rm=TRUE),

Mean\_2015 = mean(Y2015, na.rm=TRUE)) %>%

arrange(desc(Mean\_2015))

### **mutate() function**

**Use :**Creates new variables

Syntax :

mutate(data\_frame, expression(s) )

or

data\_frame %>% mutate(expression(s))

Example 28 : Create a new variable

The following code calculates division of Y2015 by Y2014 and name it "change".

mydata1 = mutate(mydata, change=Y2015/Y2014)

Example 29 : Multiply all the variables by 1000

It creates new variables and name them with suffix "\_new".

mydata11 = mutate\_all(mydata, funs("new" = .\* 1000))

|  |
| --- |
| <https://2.bp.blogspot.com/-qXG0uEVl9dY/V7iaMEImfWI/AAAAAAAAFGc/qEsiq5aeOjQg0hxpbJ9lhS-ap5m6isScwCLcB/s1600/mutate.png> |
| Output |

The output shown in the image above is truncated due to high number of variables.  
  
Note - The above code returns the following error messages -  
  
**Warning messages:**  
1: In Ops.factor(c(1L, 1L, 1L, 1L, 2L, 2L, 2L, 3L, 3L, 4L, 5L, 6L, :  
‘\*’ not meaningful for factors  
2: In Ops.factor(1:51, 1000) : ‘\*’ not meaningful for factors  
  
It implies you are multiplying 1000 to string(character) values which are stored as factor variables. These variables are 'Index', 'State'. It does not make sense to apply multiplication operation on character variables. For these two variables, it creates newly created variables which contain only NA.  
  
**Solution : See Example 45 - Apply multiplication on only numeric variables**

Example 30 : Calculate Rank for Variables

Suppose you need to calculate rank for variables Y2008 to Y2010.

mydata12 = mutate\_at(mydata, vars(Y2008:Y2010), funs(Rank=min\_rank(.)))

|  |
| --- |
| <https://2.bp.blogspot.com/-RWYQH4vMvNw/V7l6xpheaDI/AAAAAAAAFGs/-5VI-HkT0qQZ5BFWAipHSmz2UdrnI_xxQCLcB/s1600/rank%2Bmutate.png> |
| Output |

By default, **min\_rank()** assigns 1 to the smallest value and high number to the largest value. In case, you need to assign rank 1 to the largest value of a variable, use **min\_rank(desc(.))**

mydata13 = mutate\_at(mydata, vars(Y2008:Y2010), funs(Rank=min\_rank(desc(.))))

Example 31 : Select State that generated highest income among the variable 'Index'

out = mydata %>% group\_by(Index) %>% filter(min\_rank(desc(Y2015)) == 1) %>%

select(Index, State, Y2015)

Index State Y2015

1 A Alaska 1979143

2 C Connecticut 1718072

3 D Delaware 1627508

4 F Florida 1170389

5 G Georgia 1725470

6 H Hawaii 1150882

7 I Idaho 1757171

8 K Kentucky 1913350

9 L Louisiana 1403857

10 M Missouri 1996005

11 N New Hampshire 1963313

12 O Oregon 1893515

13 P Pennsylvania 1668232

14 R Rhode Island 1611730

15 S South Dakota 1136443

16 T Texas 1705322

17 U Utah 1729273

18 V Virginia 1850394

19 W Wyoming 1853858

Example 32 : Cumulative Income of 'Index' variable

The **cumsum function** calculates cumulative sum of a variable. With **mutate function,** we insert a new variable called 'Total' which contains values of cumulative income of variable Index.

out2 = mydata %>% group\_by(Index) %>% mutate(Total=cumsum(Y2015)) %>%

select(Index, Y2015, Total)

### **join() function**

**Use :** Join two datasets   
**Syntax :**

inner\_join(x, y, by = )

left\_join(x, y, by = )

right\_join(x, y, by = )

full\_join(x, y, by = )

semi\_join(x, y, by = )

anti\_join(x, y, by = )

**x, y -** datasets (or tables) to merge / join  
**by -** common variable (primary key) to join by.

Example 33 : Common rows in both the tables

df1 = data.frame(ID = c(1, 2, 3, 4, 5),

w = c('a', 'b', 'c', 'd', 'e'),

x = c(1, 1, 0, 0, 1),

y=rnorm(5),

z=letters[1:5])

df2 = data.frame(ID = c(1, 7, 3, 6, 8),

a = c('z', 'b', 'k', 'd', 'l'),

b = c(1, 2, 3, 0, 4),

c =rnorm(5),

d =letters[2:6])

**INNER JOIN** returns rows when there is a match in both tables. In this example, we are merging df1 and df2 with ID as common variable (primary key).

df3 = inner\_join(df1, df2, by = "ID")

|  |
| --- |
| <https://2.bp.blogspot.com/-WDWvZ2Ey-kk/V7mEqqBgQcI/AAAAAAAAFG8/dBgGVnxOk7w7wve3hH-RGtZv90BVBC4VgCLcB/s1600/common%2Bvar.png> |
| Output : INNER JOIN |

If the primary key does not have same name in both the tables, try the following way:

inner\_join(df1, df2, by = c("ID"="ID1"))

Example 34 : Applying LEFT JOIN

**LEFT JOIN :** It returns all rows from the left table, even if there are no matches in the right table.

left\_join(df1, df2, by = "ID")

|  |
| --- |
| <https://1.bp.blogspot.com/-WiJVqx8AyXo/V7mHu5pEjjI/AAAAAAAAFHI/Ehw1z1sRRV0dpTqUHi3KqzMHuHU38h4GwCLcB/s1600/left%2Bjoin.png> |
| Output : LEFT JOIN |

### **Combine Data Vertically**

intersect(x, y)

Rows that appear in both x and y.

union(x, y)

Rows that appear in either or both x and y.

setdiff(x, y)

Rows that appear in x but not y.

Example 35 : Applying INTERSECT

**Prepare Sample Data for Demonstration**

mtcars$model <- rownames(mtcars)

first <- mtcars[1:20, ]

second <- mtcars[10:32, ]

**INTERSECT** selects unique rows that are common to both the data frames.

intersect(first, second)

Example 36 : Applying UNION

**UNION** displays all rows from both the tables and removes duplicate records from the combined dataset. By using **union\_all function**, it allows duplicate rows in the combined dataset.

x=data.frame(ID = 1:6, ID1= 1:6)

y=data.frame(ID = 1:6, ID1 = 1:6)

union(x,y)

union\_all(x,y)

Example 37 : Rows appear in one table but not in other table

setdiff(first, second)

Example 38 : IF ELSE Statement

Syntax :

if\_else(condition, true, false, missing = NULL)

true : Value if condition meets  
false : Value if condition does not meet  
missing : Value if missing cases.It will be used to replace missing values (Default : NULL)

df <- c(-10,2, NA)

if\_else(df < 0, "negative", "positive", missing = "missing value")

Create a new variable with IF\_ELSE

If a value is less than 5, add it to 1 and if it is greater than or equal to 5, add it to 2. Otherwise 0.

df =data.frame(x = c(1,5,6,NA))

df %>% mutate(newvar=if\_else(x<5, x+1, x+2,0))

|  |
| --- |
| <https://2.bp.blogspot.com/-K4ZzNJX8E6s/V7mdNRr2gyI/AAAAAAAAFHY/SOJapQgbZ3kwoZn8deH1dvnXITu8wWqqQCLcB/s1600/ifelse.png> |
| Output |

Nested IF ELSE

Multiple IF ELSE statement can be written using if\_else() function. See the example below -

mydf =data.frame(x = c(1:5,NA))

mydf %>% mutate(newvar= if\_else(is.na(x),"I am missing",

if\_else(x==1,"I am one",

if\_else(x==2,"I am two",

if\_else(x==3,"I am three","Others")))))

Output

x flag

1 1 I am one

2 2 I am two

3 3 I am three

4 4 Others

5 5 Others

6 NA I am missing

SQL-Style CASE WHEN Statement

We can use **case\_when()** function to write nested if-else queries. In case\_when(), you can use variables directly within case\_when() wrapper. **TRUE** refers to ELSE statement.

mydf %>% mutate(flag = case\_when(is.na(x) ~ "I am missing",

x == 1 ~ "I am one",

x == 2 ~ "I am two",

x == 3 ~ "I am three",

TRUE ~ "Others"))

**Important Point**

Make sure you set is.na() condition at the beginning in nested ifelse. Otherwise, it would not be executed.

Example 39 : Apply ROW WISE Operation

Suppose you want to find maximum value in each row of variables 2012, 2013, 2014, 2015. The **rowwise()** function allows you to apply functions to rows.

df = mydata %>%

rowwise() %>% mutate(Max= max(Y2012,Y2013,Y2014,Y2015)) %>%

select(Y2012:Y2015,Max)

Example 40 : Combine Data Frames

Suppose you are asked to combine two data frames. Let's first create two sample datasets.

df1=data.frame(ID = 1:6, x=letters[1:6])

df2=data.frame(ID = 7:12, x=letters[7:12])

|  |
| --- |
| <https://1.bp.blogspot.com/-khQkXEd63R0/V7mr0aeaW2I/AAAAAAAAFH0/a2P4iNyaaaI9M8pRPEdAJiYFnYiKmVr3gCLcB/s1600/dfs.png> |
| Input Datasets |

The **bind\_rows() function** combine two datasets with rows. So combined dataset would contain **12 rows (6+6) and 2 columns.**

xy = bind\_rows(df1,df2)

It is equivalent to base R function rbind.

xy = rbind(df1,df2)

The **bind\_cols() function** combine two datasets with columns. So combined dataset would contain **4 columns and 6 rows.**

xy = bind\_cols(x,y)

or

xy = cbind(x,y)

The output is shown below-

|  |
| --- |
| <https://4.bp.blogspot.com/-FH7CRI-GBxk/V7mtWzG0KDI/AAAAAAAAFIA/QnSNd_yeUi4Tj1K9V3HyrHo92pJIOspUgCLcB/s1600/cbind.png> |
| cbind Output |

Example 41 : Calculate Percentile Values

The **quantile()** function is used to determine Nth percentile value. In this example, we are computing percentile values by variable Index.

mydata %>% group\_by(Index) %>%

summarise(Pecentile\_25=quantile(Y2015, probs=0.25),

Pecentile\_50=quantile(Y2015, probs=0.5),

Pecentile\_75=quantile(Y2015, probs=0.75),

Pecentile\_99=quantile(Y2015, probs=0.99))

The **ntile()** function is used to divide the data into N bins.

x= data.frame(N= 1:10)

x = mutate(x, pos = ntile(x$N,5))

Example 42 : Automate Model Building

This example explains the advanced usage of **do() function**. In this example, we are building linear regression model for each level of a categorical variable. There are 3 levels in variable cyl of dataset mtcars.

length(unique(mtcars$cyl))

**Result : 3**

by\_cyl <- group\_by(mtcars, cyl)

models <- by\_cyl %>% do(mod = lm(mpg ~ disp, data = .))

summarise(models, rsq = summary(mod)$r.squared)

models %>% do(data.frame(

var = names(coef(.$mod)),

coef(summary(.$mod)))

)

|  |
| --- |
| <https://4.bp.blogspot.com/-Td-hA3wRNFE/V7m3ONwn-zI/AAAAAAAAFIc/DUQoz6SGmFs1qd05J6PpDs6sxmcHLs4VACLcB/s1600/rsquare.png> |
| Output : R-Squared Values |

### **if() Family of Functions**

It includes functions like select\_if, mutate\_if, summarise\_if. They come into action only when logical condition meets. See examples below.

Example 43 : Select only numeric columns

The **select\_if()** function returns only those columns where logical condition is TRUE. The **is.numeric** refers to retain only numeric variables.

mydata2 = select\_if(mydata, is.numeric)

Similarly, you can use the following code for selecting factor columns -

mydata3 = select\_if(mydata, is.factor)

Example 44 : Number of levels in factor variables

Like select\_if() function, summarise\_if() function lets you to summarise only for variables where logical condition holds.

summarise\_if(mydata, is.factor, funs(nlevels(.)))

It returns 19 levels for variable Index and 51 levels for variable State.

Example 45 : Multiply by 1000 to numeric variables

mydata11 = mutate\_if(mydata, is.numeric, funs("new" = .\* 1000))

Example 46 : Convert value to NA

In this example, we are converting "" to NA using **na\_if()** function.

k <- c("a", "b", "", "d")

na\_if(k, "")

**Result :** "a" "b" NA "d"

Example 47 : Use of pull( ) function

iris %>% pull(Sepal.Length) is equivalent to writing iris$Sepal.Length or iris[["Sepal.Length"]] If you want output to be in vector rather than data frame (default method), you can use pull( ) function.

iris %>% filter(Sepal.Length > 5.5) %>% pull(Species)

Example 48 : How to deal with Quotation

Let's understand with example. You want to use a variable which is in quotes. In the example below, **Species** is in quotes. If you use quoted variable directly, it would return zero rows. To make it work, you need to use !! operator which unquotes its argument and gets evaluated immediately in the surrounding context. The final thing we need to do is turn the character string "Species" into Species, a symbol by using sym function.

filter\_df <- function(df, colname, val){

filter(df, colname == val)

}

filter\_df(iris,"Species", "setosa")

Output

Zero rows

filter\_df <- function(df, colname, val){

filter(df, !!sym(colname) == val)

}

filter\_df(iris,"Species", "setosa")

Output

50 rows

enquo() is used to quote its argument. Here we are asking user to define variable name without quotes.

filter\_df <- function(df, colname, val){

colname = enquo(colname)

filter(df, !!colname == val)

}

filter\_df(iris, Species, "setosa")

Example 49 : How to use SQL rank() over(partition by)

In SQL, rank() over(partition by) is used to compute rank by a grouping variable. In dplyr, it can be achieved very easily with a single line of code. See the example below. Here we are calculating rank of variable Y2015 by variable Index.

t = mydata %>% select(Index, Y2015) %>%

group\_by(Index) %>%

mutate(rank = min\_rank(desc(Y2015)))%>%

arrange(Index, rank)

In dplyr, there are many functions to compute rank other than min\_rank( ). These are dense\_rank( ), row\_number( ), percent\_rank().

### **across() function**

**across( )** function was added starting dplyr version 1.0. It helps analyst to perform same operation on multiple columns. Let's take a sample data.frame mtcars and calculate mean on variables from 'mpg' through 'qsec' by 'carb'.

Alternative to summarise\_at function

mtcars %>%

group\_by(carb) %>%

summarise(across(mpg:qsec, mean))

Alternative to summarise\_if function

The code below calculates average on numeric variables. It identifies numeric variables using where() function.

mtcars %>%

group\_by(carb) %>%

summarise(across(where(is.numeric), mean))

Multiple across() function

Here we are using two summary statistics - mean and no. of distinct values in two different set of variables.

mtcars %>%

group\_by(carb) %>%

summarise(across(mpg:qsec, mean), across(vs:gear, n\_distinct))

across() can also be applied with mutate function

mtcars %>%

group\_by(carb) %>%

mutate(across(where(is.numeric), mean))

**Some other examples of across() function -**

df %>% mutate(across(c(x, starts\_with("y")), mean, na.rm = TRUE))   
df %>% mutate(across(everything(), mean, na.rm = TRUE))

## What are valid variable names in R?

A valid variable name consists of letters, numbers and the dot or underline characters. A variable name can start with either a letter or the dot followed by a character **(not number)**.

A variable name such as **.1var** is not valid. But .var1 is valid.

A variable name cannot have reserved words. The reserved words are listed below -

if else repeat while function for in next break  
TRUE FALSE NULL Inf NaN NA NA\_integer\_ NA\_real\_ NA\_complex\_ NA\_character\_

## What is the use of with() and by() functions? What are its alternatives?

Suppose you have a data frame as shown below -

df=data.frame(x=c(1:6), y=c(1,2,4,6,8,12))

You are asked to perform this calculation : **(x+y) + (x-y) .** Most of the R programmers write like code below -

(df$x + df$y) + (df$x - df$y)

Using **with() function**, you can refer your data frame and make the above code compact and simpler-

with(df, (x+y) + (x-y))

The with() function is equivalent to pipe operator in dplyr package. See the code below -

library(dplyr)  
df %>% mutate((x+y) + (x-y))

**by() function in R**  
  
The by() function is equivalent to **group by function**in SQL. It is used to perform calculation by a factor or a categorical variable. In the example below, we are computing mean of variable var2 by a factor var1.

df = data.frame(var1=factor(c(1,2,1,2,1,2)), var2=c(10:15))  
with(df, by(df, var1, function(x) mean(x$var2)))

The **group\_by() function** in dply package can perform the same task.

library(dplyr)  
df %>% group\_by(var1)%>% summarise(mean(var2))

## 15. How to rename a variable?

In the example below, we are renaming variable var1 to variable1.

df = data.frame(var1=c(1:5))  
colnames(df)[colnames(df) == 'var1'] <- 'variable1'

The **rename()** function in dplyr package can also be used to rename a variable.

library(dplyr)  
df= rename(df, variable1=var1)

## 16. What is the use of which() function in R?

The **which() function** returns the position of elements of a logical vector that are TRUE. In the example below, we are figuring out the row number wherein the maximum value of a variable x is recorded.

mydata=data.frame(x = c(1,3,10,5,7))  
**which**(mydata$x==max(mydata$x))

It returns 3 as 10 is the maximum value and it is at 3rd row in the variable x.

## 17. How to calculate first non-missing value in variables?

Suppose you have three variables X, Y and Z and you need to extract first non-missing value in each rows of these variables.

data = read.table(text="  
X Y Z  
NA 1 5  
3 NA 2  
", header=TRUE)

The **coalesce()** function in dplyr package can be used to accomplish this task.

library(dplyr)  
data %>% mutate(var=**coalesce**(X,Y,Z))

|  |
| --- |
| <https://4.bp.blogspot.com/-tCRz4mjKHJc/V8sw4CJAgPI/AAAAAAAAFUU/xfUCI1xpcgYTljmvq4e9ZFPh_Vt_t6MggCLcB/s1600/coalesce.png> |
| COALESCE Function in R |

## How to calculate max value for rows?

*Let's create a sample data frame*

dt1 = read.table(text="  
X Y Z  
7 NA 5  
2 4 5  
", header=TRUE)

With **apply() function**, we can tell R to apply the max function rowwise. The **na,rm = TRUE** is used to tell R to ignore missing values while calculating max value. If it is not used, it would return NA.

dt1$var = apply(dt1,1, function(x) max(x,na.rm = TRUE))

|  |
| --- |
| <https://1.bp.blogspot.com/-xLFqknSpUnE/V8vhiksGWGI/AAAAAAAAFUk/wDNFVjkr-pU6CaZTI9hkKMVg9ys7xg7BwCLcB/s1600/max%2Brow.png> |
| Output |

## 19. Count number of zeros in a row

dt2 = read.table(text="  
A B C  
8 0 0  
6 0 5  
", header=TRUE)

apply(dt2,1, function(x) sum(x==0))

## 20. Does the following code work?

ifelse(df$var1==NA, 0,1)

It does not work. The logic operation on NA returns NA. It does not TRUE or FALSE.  
  
This code works **ifelse(is.na(df$var1), 0,1)**

## 21. What would be the final value of x after running the following program?

x = 3  
mult <- function(j)  
{  
  x = j \* 2  
  return(x)  
}  
  
mult(2)  
[1] 4  
  
**Answer :** The value of 'x' will remain 3. See the output shown in the image below-

|  |
| --- |
| <https://4.bp.blogspot.com/-yZzcLAR-59Y/V8vpSKyv_8I/AAAAAAAAFU0/7WBesVUx-9kNkDh5-MuGRtcqiPe7BSlAgCLcB/s1600/func.png> |
| Output |

It is because x is defined outside function. If you want to change the value of x after running the function, you can use the following program:

x = 3  
mult <- function(j)  
{  
  x <<- j \* 2  
  return(x)  
}  
mult(2)  
x

The operator "<<-" tells R to search in the parent environment for an existing definition of the variable we want to be assigned.

## 22. How to convert a factor variable to numeric

The as.numeric() function returns a vector of the levels of your factor and not the original values. Hence, it is required to convert a factor variable to character before converting it to numeric.

a <- factor(c(5, 6, 7, 7, 5))  
a1 = as.numeric(as.character(a))

## 23. How to concatenate two strings?

The **paste() function** is used to join two strings. A single space is the default separator between two strings.

a = "Deepanshu"  
b = "Bhalla"  
paste(a, b)

It returns "Deepanshu Bhalla"  
  
If you want to change the default single space separator, you can add sep="," keyword to include comma as a separator.

paste(a, b, sep=",") returns "Deepanshu,Bhalla"

## 24. How to extract first 3 characters from a word

The substr() function is used to extract strings in a character vector. The syntax of substr function is **substr(character\_vector, starting\_position, end\_position)**

x = "AXZ2016"  
substr(x,1,3)

## 25. How to extract last name from full name

The last name is the end string of the name. For example, Jhonson is the last name of "Dave,Jon,Jhonson".

dt2 = read.table(text="  
var  
Sandy,Jones  
Dave,Jon,Jhonson  
", header=TRUE)

The word() function of stringr package is used to extract or scan word from a string. -1 in the second parameter  denotes the last word.

library(stringr)  
dt2$var2 = word(dt2$var, -1, sep = ",")

## 26. How to remove leading and trailing spaces

The **trimws()** function is used to remove leading and trailing spaces.

a = " David Banes "  
trimws(a)

It returns "David Banes".

## 27. How to generate random numbers between 1 and 100

The runif() function is used to generate random numbers.

rand = runif(100, min = 1, max = 100)

## 28. How to apply LEFT JOIN in R?

**LEFT JOIN** implies keeping all rows from the left table (data frame) with the matches rows from the right table. In the merge() function, all.x=TRUE denotes left join.

df1=data.frame(ID=c(1:5), Score=runif(5,50,100))  
df2=data.frame(ID=c(3,5,7:9), Score2=runif(5,1,100))  
comb = **merge(df1, df2, by ="ID", all.x = TRUE)**

**Left Join (SQL Style)**

library(sqldf)  
comb = sqldf('select df1.\*, df2.\* from df1 left join df2 on df1.ID = df2.ID')

**Left Join with dply package**

library(dplyr)  
comb = left\_join(df1, df2, by = "ID")

[**Joining and Merging with R**](http://www.listendata.com/2015/08/joining-and-merging-in-r.html)

## 29. How to calculate cartesian product of two datasets

The cartesian product implies cross product of two tables (data frames). For example, df1 has 5 rows and df2 has 5 rows. The combined table would contain 25 rows (5\*5)

comb = merge(df1,df2,by=NULL)

**CROSS JOIN (SQL Style)**

library(sqldf)  
comb2 = sqldf('select \* from df1 join df2 ')

## 30. Unique rows common to both the datasets

*First, create two sample data frames*  
  
df1=data.frame(ID=c(1:5), Score=c(50:54))  
df2=data.frame(ID=c(3,5,7:9), Score=c(52,60:63))

library(dplyr)  
comb = intersect(df1,df2)

library(sqldf)  
comb2 = sqldf('select \* from df1 intersect select \* from df2 ')

|  |
| --- |
| <https://4.bp.blogspot.com/-pDzg_KQIp38/V8xHxTRZy0I/AAAAAAAAFVE/aC27xWKcuwQDmfvmwqwNBdIviHdf_Jv7gCLcB/s1600/intersection.png> |
| Output : Intersection with R |

## 31. How to measure execution time of a program in R?

There are multiple ways to measure running time of code. Some frequently used methods are listed below -  
 **R Base Method**

start.time <- Sys.time()  
runif(5555,1,1000)  
end.time <- Sys.time()  
end.time - start.time

**With tictoc package**

library(tictoc)  
tic()  
runif(5555,1,1000)  
toc()

## 32. Which package is generally used for fast data manipulation on large datasets?

The package **data.table** performs fast data manipulation on large datasets. See the comparison between dplyr and data.table.

# Load data  
library(nycflights13)  
data(flights)  
df = setDT(flights)

# Load required packages  
library(tictoc)  
library(dplyr)  
library(data.table)  
  
# Using data.table package  
tic()  
df[arr\_delay > 30 & dest == "IAH",  
   .(avg = mean(arr\_delay),  
     size = .N),  
   by = carrier]  
toc()  
  
# Using dplyr package  
tic()  
flights %>% filter(arr\_delay > 30 & dest == "IAH") %>%  
  group\_by(carrier) %>% summarise(avg = mean(arr\_delay), size = n())  
toc()

**Result :** data.table package took 0.04 seconds. whereas dplyr package took 0.07 seconds. So, data.table is approx. 40% faster than dplyr. Since the dataset used in the example is of medium size, there is no noticeable difference between the two. As size of data grows, the difference of execution time gets bigger.

## 33. How to read large CSV file in R?

We can use **fread()** function of data.table package.

library(data.table)  
yyy = fread("C:\\Users\\Dave\\Example.csv", header = TRUE)

We can also use **read.big.matrix()** function of bigmemory package.

## 34. What is the difference between the following two programs ?

1. temp = data.frame(v1<-c(1:10),v2<-c(5:14))  
2. temp = data.frame(v1=c(1:10),v2=c(5:14))

In the first case, it created two vectors v1 and v2 and a data frame temp which has 2 variables with improper variable names. The second code creates a data frame temp with proper variable names.

## 35. How to remove all the objects

rm(list=ls())

## 36. What are the various sorting algorithms in R?

Major five sorting algorithms :

1. Bubble Sort
2. Selection Sort
3. Merge Sort
4. Quick Sort
5. Bucket Sort

## 37. Sort data by multiple variables

**Create a sample data frame**

mydata = data.frame(score = ifelse(sign(rnorm(25))==-1,1,2),  
                    experience= sample(1:25))

**Task :**You need to sort score variable on ascending order and then sort experience variable on **descending** order.

**R Base Method**

mydata1 <- mydata[order(mydata$score, -mydata$experience),]

**With dplyr package**

library(dplyr)  
mydata1 = arrange(mydata, score, **desc**(experience))

## 38. Drop Multiple Variables

Suppose you need to remove 3 variables - x, y and z from data frame "mydata".

**R Base Method**

df = subset(mydata, select = -c(x,y,z))

**With dplyr package**

library(dplyr)  
df = select(mydata, -c(x,y,z))

## 40. How to save everything in R session

save.image(file="dt.RData")

## 41. How R handles missing values?

Missing values are represented by capital NA.  
  
To create a new data without any missing value, you can use the code below :

df <- na.omit(mydata)

## 42. How to remove duplicate values by a column

Suppose you have a data consisting of 25 records. You are asked to remove duplicates based on a column. In the example, we are eliminating duplicates by variable y.

data = data.frame(y=sample(1:25, replace = TRUE), x=rnorm(25))

**R Base Method**

test = subset(data, !duplicated(data[,"y"]))

**dplyr Method**

library(dplyr)  
test1 = distinct(data, y, .keep\_all= TRUE)

## 43. Which packages are used for transposing data with R

The reshape2 and tidyr packages are most popular packages for reshaping data in R.  
  
[**Explanation : Transpose Data**](http://www.listendata.com/2016/01/transpose-data-in-r.html)

## 44. Calculate number of hours, days, weeks, months and years between 2 dates

*Let's set 2 dates :*

dates <- as.Date(c("2015-09-02", "2016-09-05"))

difftime(dates[2], dates[1], units = "hours")  
difftime(dates[2], dates[1], units = "days")  
floor(difftime(dates[2], dates[1], units = "weeks"))  
floor(difftime(dates[2], dates[1], units = "days")/365)

**With lubridate package**

library(lubridate)  
interval(dates[1], dates[2]) %/% hours(1)  
interval(dates[1], dates[2]) %/% days(1)  
interval(dates[1], dates[2]) %/% weeks(1)  
interval(dates[1], dates[2]) %/% months(1)  
interval(dates[1], dates[2]) %/% years(1)

The number of months unit is not included in the base difftime() function so we can use interval() function of lubridate() package.

## 45. How to add 3 months to a date

mydate <- as.Date("2015-09-02")  
mydate + months(3)

## 46. Extract date and time from timestamp

mydate <- as.POSIXlt("2015-09-27 12:02:14")  
library(lubridate)  
date(mydate) # Extracting date part  
format(mydate, format="%H:%M:%S") # Extracting time part

*Extracting various time periods*

day(mydate)  
month(mydate)  
year(mydate)  
hour(mydate)  
minute(mydate)  
second(mydate)

## 47. What are various ways to write loop in R

There are primarily three ways to write loop in R

1. For Loop
2. While Loop
3. Apply Family of Functions such as Apply, Lapply, Sapply etc

## 48. Difference between lapply and sapply in R

lapply returns a list when we apply a function to each element of a data structure. whereas sapply returns a vector.

## 49. Difference between sort(), rank() and order() functions?

The sort() function is used to sort a 1 dimension vector or a single variable of data.

The rank() function returns the ranking of each value.

The order() function returns the indices that can be used to sort the data.

**Example :**

set.seed(1234)  
x = sample(1:50, 10)     
x

[1]  6 31 30 48 40 29  1 10 28 22

**sort(x)**

[1]  1  6 10 22 28 29 30 31 40 48

*It sorts the data on ascending order.*

**rank(x)**

[1]  2  8  7 10  9  6  1  3  5  4

2 implies the number in the first position is the second lowest and 8 implies the number in the second position is the eighth lowest.

**order(x)**

 [1]  7  1  8 10  9  6  3  2  5  4

7 implies the 7th value of x is the smallest value, so 7 is the first element of order(x) and i refers to the first value of x is the second smallest.

If you run **x[order(x)]**, it would give you the same result as sort() function. The difference between these two functions lies in two or more dimensions of data (two or more columns). In other words, the sort() function cannot be used for more than 1 dimension whereas x[order(x)] can be used.

## 50.  Extracting Numeric Variables

cols <- sapply(mydata, is.numeric)  
abc = mydata [,cols]

**Data Science with R Interview Questions**  
  
The list below contains most frequently asked interview questions for a role of data scientist. Most of the roles related to data science or predictive modeling require candidate to be well conversant with R and know how to develop and validate predictive models with R.

## 51. Which function is used for building linear regression model?

The lm() function is used for fitting a linear regression model.

## 52. How to add interaction in the linear regression model?

:An interaction can be created using colon sign (:). For example, x1 and x2 are two predictors (independent variables). The interaction between the variables can be formed like **x1:x2.**  
**See the example below -**

linreg1 <- lm(y ~ x1 + x2 + x1:x2, data=mydata)

The above code is equivalent to the following code :

linreg1 <- lm(y ~ x1\*x2, data=mydata)

**x1:x2 -** It implies including both main effects (x1 + x2) and interaction (x1:x2).

## 53. How to check autocorrelation assumption for linear regression?

durbinWatsonTest() function

## 54. Which function is useful for developing a binary logistic regression model?

glm() function with family = "binomial"

## 55. How to perform stepwise variable selection in logistic regression model?

Run step() function after building logistic model with glm() function.

## 56. How to do scoring in the logistic regression model?

Run predict(logit\_model, validation\_data, type = "response")

## 57. How to split data into training and validation?

dt = sort(sample(nrow(mydata), nrow(mydata)\*.7))  
train<-mydata[dt,]  
val<-mydata[-dt,]

## 58. How to standardize variables?

data2 = scale(data)

## 60. Which are the popular R packages for decision tree?

rpart, party

## 61. What is the difference between rpart and party package for developing a decision tree model?

rpart is based on Gini Index which measures impurity in node. Whereas ctree() function from "party" package uses a significance test procedure in order to select variables.

## 62. How to check correlation with R?

cor() function

## 63. Have you heard 'relaimpo' package?

It is used to measure the relative importance of independent variables in a model.

## 64. How to fine tune random forest model?

Use tuneRF() function

## 65. What shrinkage defines in gradient boosting model?

Shrinkage is used for reducing, or shrinking, the impact of each additional fitted base-learner (tree).

## 66. How to make data stationary for ARIMA time series model?

Use ndiffs() function which returns the number of difference required to make data stationary.

## 67. How to automate arima model?

Use auto.arima() function of forecast package

<https://www.listendata.com/2016/08/dplyr-tutorial.html>

# *****Examples of Quant interview questions*****

## Can you calculate the delta of an option without using the precise formula for delta (the variables are given).

I guess one need to use Taylor's series to bypass the N(.) i.e. the cumulative distribution calculation.

**ANSWER:** <https://math.stackexchange.com/questions/3897740/delta-of-options-a-mathematical-explanation>

[](https://www.glassdoor.com/Interview/Explain-your-research-for-4-times-QTN_1820415.htm)

## How to reduce dimension when you do regression?

**ANSWER:** PCA, exhaustive search

<https://stats.stackexchange.com/questions/20836/algorithms-for-automatic-model-selection>

## What is Brownian motion/Martingale?

**ANSWER:** Brownian motion – stochastic process, which is:

* 1. Continuous
  2. Nondifferentiable
  3. Has stationary increments, which are iid normally distributed with 0 mean and const variance
  4. Without jumps

Martingale – a process, where expected value at t + 1 is equal to value at t, given filtration F(t) (no dependency pre t)

## Give an example that is a Martingale but without Markov property.

**ANSWER:**

<https://math.stackexchange.com/questions/503245/stochastic-process-that-is-martingale-but-not-markov>

[](https://www.glassdoor.com/Interview/Structures-lists-testing-of-python-programming-and-some-SVM-sklearn-questions-Also-some-questions-on-non-clustered-and-QTN_2472149.htm)

[Market Risk Quantitative Analyst was asked...](https://www.glassdoor.com/Interview/Structures-lists-testing-of-python-programming-and-some-SVM-sklearn-questions-Also-some-questions-on-non-clustered-and-QTN_2472149.htm)February 5, 2018

## Structures, lists, testing of python programming and some SVM, sklearn questions. Also some questions on non-clustered and clustered data in SQL.

One need strong python skills and they will understand your background from some projects that you have done already and then will start asking you questions.

**ANSWERS:**

* **Structures**

Answer here 15.1

* **Lists**

Answer here 15.1

* **Python code testing**
* **SVM**

Support vector machines (SVMs) are a set of supervised learning methods used for classification, regression and outliers detection.

The **advantages** of support vector machines are

* Effective in high dimensional spaces.
* Still effective in cases where number of dimensions is greater than the number of samples.
* Uses a subset of training points in the decision function (called support vectors), so it is also memory efficient.
* Versatile: different Kernel functions can be specified for the decision function. Common kernels are provided, but it is also possible to specify custom kernels.

The **disadvantages** of support vector machines include:

* If the number of features is much greater than the number of samples, avoid over-fitting in choosing Kernel functions and regularization term is crucial.
* SVMs do not directly provide probability estimates, these are calculated using an expensive five-fold cross-validation (see Scores and probabilities, below).
* **Sklearn**

The support vector machines in scikit-learn support both dense (numpy.ndarray and convertible to that by numpy.asarray) and sparse (any scipy.sparse) sample vectors as input. However, to use an SVM to make predictions for sparse data, it must have been fit on such data. For optimal performance, use C-ordered numpy.ndarray (dense) or scipy.sparse.csr\_matrix (sparse) with dtype=float64.

* **Clustered/non-clustered data in SQL**

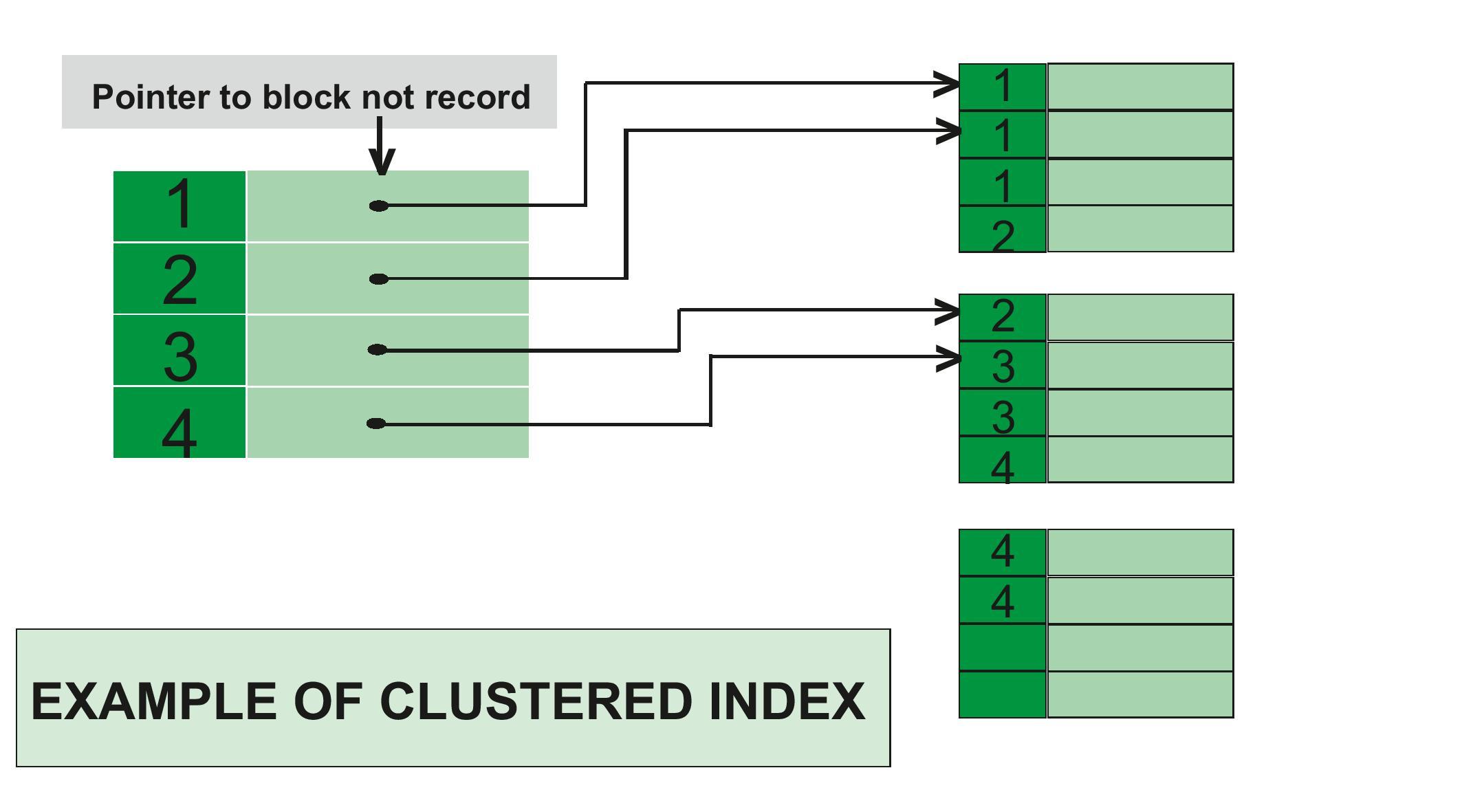
**1. Clustered Index :**   
Clustered index is created only when both the following conditions satisfy –

The data or file, that you are moving into secondary memory should be in sequential or sorted order.

There should be a key value, meaning it cannot have repeated values. 

Whenever you apply clustered indexing in a table, it will perform sorting in that table only. You can create only one clustered index in a table like primary key. Clustered index is as same as dictionary where the data is arranged by alphabetical order.

In clustered index, index contains pointer to block but not direct data. 



**Example of Clustered Index –**   
If you apply primary key to any column, then automatically it will become clustered index. 

create table Student

( Roll\_No int primary key,

Name varchar(50),

Gender varchar(30),

Mob\_No bigint );

insert into Student

values (4, 'ankita', 'female', 9876543210 );

insert into Student

values (3, 'anita', 'female', 9675432890 );

insert into Student

values (5, 'mahima', 'female', 8976453201 );

In this example, Roll no is a primary key, it will automatically act as a clustered index.   
The output of this code will produce in increasing order of roll no.

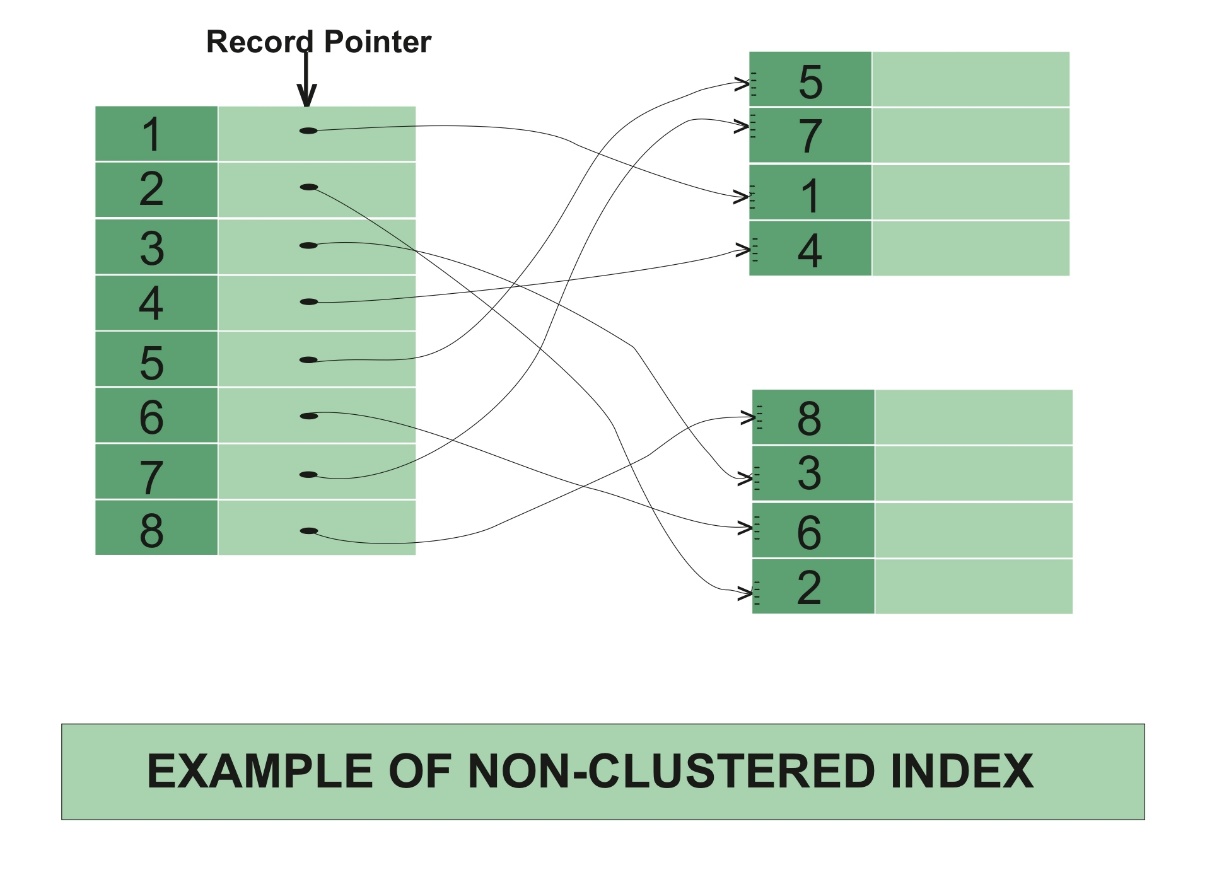
| Roll\_No | Name | Gender | Mob\_No |
| --- | --- | --- | --- |
| 3 | anita | female | 9675432890 |
| 4 | ankita | female | 9876543210 |
| 5 | mahima | female | 8976453201 |

You can have only one clustered index in one table, but you can have one clustered index on multiple columns, and that type of index is called composite index.

**2. Non-clustered Index :**   
Non-Clustered Index is similar to the index of a book. The index of a book consists of a chapter name and page number, if you want to read any topic or chapter then you can directly go to that page by using index of that book. No need to go through each and every page of a book.

The data is stored in one place, and index is stored in another place. Since, the data and non-clustered index is stored separately, then you can have multiple non-clustered index in a table.

In non-clustered index, index contains the pointer to data.



**Example of Non-clustered Index –**

create table Student

( Roll\_No int primary key,

Name varchar(50),

Gender varchar(30),

Mob\_No bigint );

insert into Student

values (4, 'afzal', 'male', 9876543210 );

insert into Student

values (3, 'sudhir', 'male', 9675432890 );

insert into Student

values (5, 'zoya', 'female', 8976453201 );

create nonclustered index NIX\_FTE\_Name

on Student (Name ASC);

Here, roll no is a primary key, hence there is automatically a clustered index.   
If we want to apply non-clustered index in NAME column (in ascending order), then the new table will be created for that column.

Output before applying non-clustered index : 

| Roll\_No | Name | Gender | Mob\_No |
| --- | --- | --- | --- |
| 3 | sudhir | male | 9675432890 |
| 4 | afzal | male | 9876543210 |
| 5 | zoya | female | 8976453201 |

Output after applying non-clustered index :

| Name | Row address |
| --- | --- |
| Afzal | 3452 |
| Sudhir | 5643 |
| zoya | 9876 |

Row address is used because, if someone wants to search the data for sudhir, then by using the row address he/she will directly go to that row address and can fetch the data directly.

**Difference between Clustered and Non-clustered index :**

| **CLUSTERED INDEX** | **NON-CLUSTERED INDEX** |
| --- | --- |
| Clustered index is faster. | Non-clustered index is slower. |
| Clustered index requires less memory for operations. | Non-Clustered index requires more memory for operations. |
| In clustered index, index is the main data. | In Non-Clustered index, index is the copy of data. |
| A table can have only one clustered index. | A table can have multiple non-clustered index. |
| Clustered index has inherent ability of storing data on the disk. | Non-Clustered index does not have inherent ability of storing data on the disk. |
| Clustered index store pointers to block not data. | Non-Clustered index store both value and a pointer to actual row that holds data. |
| In Clustered index leaf nodes are actual data itself. | In Non-Clustered index leaf nodes are not the actual data itself rather they only contains included columns. |
| In Clustered index, Clustered key defines order of data within table. | In Non-Clustered index, index key defines order of data within index. |
| A Clustered index is a type of index in which table records are physically reordered to match the index. | A Non-Clustered index is a special type of index in which logical order of index does not match physical stored order of the rows on disk. |
| The size of clustered index is large. | Size of non-clustered index is comparatively smaller. |
| Primary Keys of the table by default are clustered index. | Composite key when used with unique constraints of the table act as non-clustered index. |

[](https://www.glassdoor.com/Interview/What-is-VaR-in-simple-terms-and-how-would-you-explain-it-to-someone-who-never-heard-of-it-What-is-a-martingale-in-a-detail-QTN_1559595.htm)

## What is VaR in simple terms and how would you explain it to someone who never heard of it.

Value-at-Risk – it is a risk measure, which returns the maximum loss generated by a portfolio in a given time period, give a predefined confidence level (with a predefined probability). It does not explain what is the maximum loss possible or in general what is the distribution of losses beyond the confidence level.

## What is a martingale in a detailed way.

An additional property that holds for our sequence of partial sums is the **Martingale property**. It states **that the conditional expectation of the sequence of partial sums, Si is simply the current value**:

*E(Si|*Sk,k<i)=Sk

Essentially, the martingale property ensures that in a "fair game", knowledge of the past will be of no use in predicting future winnings.

## Questions about stochastic models that are used, what are their differences.

* Monte Carlo Simulation
* Brownian Motion / Wiener process -> normal distribution, stationary increments, starts at 0, continuous
* Arithmetic Brownian Motion -> can be negative
* Geometric Brownian Motion -> always positive, may be explosive
* Poisson process
* Brownian motion with jumps
* Merton Model
* Ornstein-Uhlenbeck process -> mean-reverting (e.g. Vasicek, Cox-Ingersoll-Ross)

[](https://www.glassdoor.com/Interview/Expectation-and-Variance-of-Brownian-motion-multiplied-by-sqrt-of-t-QTN_4556727.htm)

## Expectation and Variance of Brownian motion multiplied by sqrt of t

X(t) – Brownian motion; E(X(t)) = 0, Var(X(t)) = t

Z(t) = sqrt(t) \* X(t)

E(Z(t)) = sqrt(t) \* 0 = 0

Var(X(t)) = t \* Var(X(t)) = t\* t = t^2

[](https://www.glassdoor.com/Interview/Interview-1-Walk-me-through-your-CV-Technical-questions-on-probability-distributions-normal-uniform-etc-VaR-ES-Arit-QTN_4430719.htm)

**Interview 1:**

## Walk me through your CV.

* SGH / Tilburg
* Deloitte – Actuarial cashflow models, ALM (ALS), generating paths of Economic scenarios (interest rates), coded in SQL/VBA
* CS – coded in R, stress testing models implementation/development (Cash Equities, Prime Brokerage, Corporate Equity Derivatives – collars/call spreads, Structured Equity Derivatives – Exotics, Structured Financing, Securitized Products)
* TripleA- scenario generation, Python cashflow shadow model, implementation of cashflow models in R
* JPMC – stress testing development of NIR (IB Fees, Expenses), NII models, time series decomposition (ARIMA, seasonality etc), Economic scenario generation of US UER

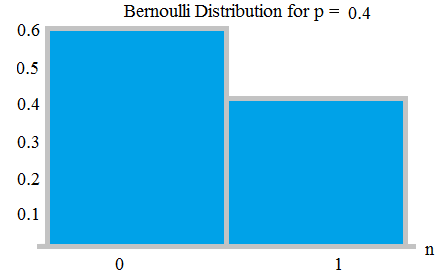
## Technical questions on probability distributions (normal, uniform etc),

**Some interview questions based on probability distributions frequently asked in interviews:**

1. Given a random generator that produces a number 1 to 5 uniformly, write a function that produces a number from 1 to 7 uniformly.
2. Three friends in Seattle told you it’s rainy. Each has a probability of 1/3 of lying. What’s the probability that Seattle is rainy?
3. There are 6 marbles in a bag — 1 is white. You reach in the bag 100 times. After drawing a marble, it is placed back in the bag. What is the probability of drawing that white marble at least once?
4. How to write a function to make a biased coin from a fair coin?
5. Estimate the disease probability in one city given the probability is very low nationwide. We randomly asked 1000 people in this city, with all negative response(NO disease). What is the probability of disease in the city?

**Bernoulli and Uniform**

When there is a tossing of a coin, we think of [Bernoulli’s distribution](https://en.wikipedia.org/wiki/Bernoulli_distribution). It represents a [coin toss](https://en.wikipedia.org/wiki/Coin_toss) where 1 and 0 would represent “heads” and “tails” (or vice versa), respectively, and p would be the probability of the coin landing on heads or tails, respectively. The outcome of the experiment is boolean in nature. For example, the probability of getting a heads while flipping a coin is 0.5. The probability of tails is 1 — P (1 minus the probability of heads, which also equals 0.5 for a coin toss).



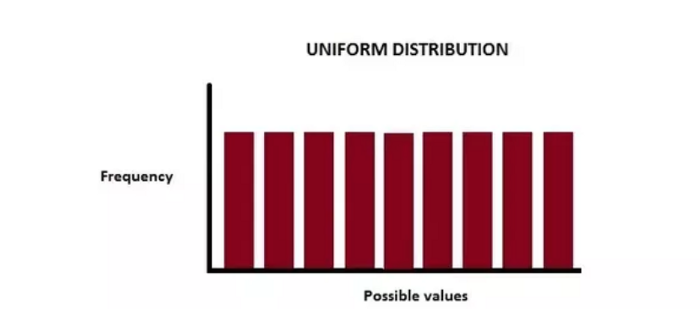
Source: [Statistics How To](https://www.statisticshowto.datasciencecentral.com/bernoulli-distribution/)

Probability Density function(PDF) for Bernoulli’s distribution:

https://miro.medium.com/max/157/0*EF7sRIxXUYizd-0A.gif

Source: [Wolfram](http://mathworld.wolfram.com/BernoulliDistribution.html)

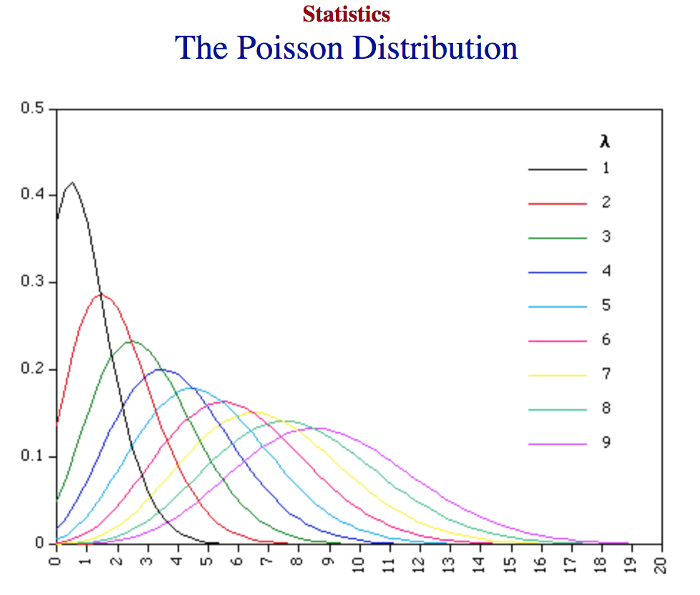
To learn more about Bernoulli’s distribution, like mean, standard deviation and variance, please visit [wolfram mathworld](http://mathworld.wolfram.com/BernoulliDistribution.html).



Imagine rolling a fair die. The outcomes 1 to 6 are equally likely. It can be defined for any number of outcomes n or even as a continuous distribution. This is [uniform distribution](https://en.wikipedia.org/wiki/Uniform_distribution_(discrete)), characterized by its flat PDF. A uniform distribution, is also called a rectangular distribution, is a probability distribution that has constant probability.

**Poisson**

How would you model the count of customers clicking a particular link on your website each minute? The entire number of clicks in one day are modelled by a Poisson distribution. Poisson Distribution is applicable in situations where events occur at random bursts of time and space and our interest lies only in the total number of occurrences of the event. The limiting result is the [Poisson distribution](https://en.wikipedia.org/wiki/Poisson_distribution).



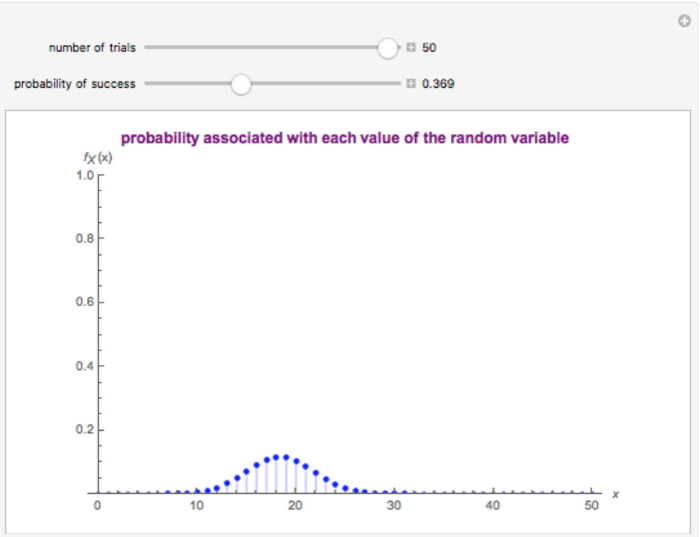
Source: [UMass Website](https://www.umass.edu/wsp/resources/poisson/)

Where the rate of occurrence of some event, **r** (in this chart called lambda or **l**) is small, the range of likely possibilities will lie near the zero line. Meaning that when the rate **r** is small, zero is a very likely number to get. As the rate becomes higher (as the occurrence of the thing we are watching becomes more common), the center of the curve moves toward the right, and eventually, somewhere around when **r** = 7, zero occurrences will actually become unlikely.

When customers arrive at a movie theatre, to buy a ticket First Come First Serve(FIFO), the distribution is similar to a Poisson distribution.

For more detailed reading: [The Poisson Distribution](https://www.umass.edu/wsp/resources/poisson/)

**Binomial**



Source: <http://demonstrations.wolfram.com/BinomialProbabilityDistribution/>

Summation of outcomes of a Bernoulli’s distribution is a Binomial distribution. When you toss a coin more than once and want to map the outcome, we use this distribution. When tossing the coin n times, count is an outcome that follows the binomial distribution. Its parameters are n, the number of trials, and p, the probability of a “success” (maybe heads). Each flip is a Bernoulli’s [trial](https://en.wikipedia.org/wiki/Bernoulli_trial). Here, it should be noted that the flip of each coin is independent of the other flips.

The question 3 above, the count also follows a binomial distribution.

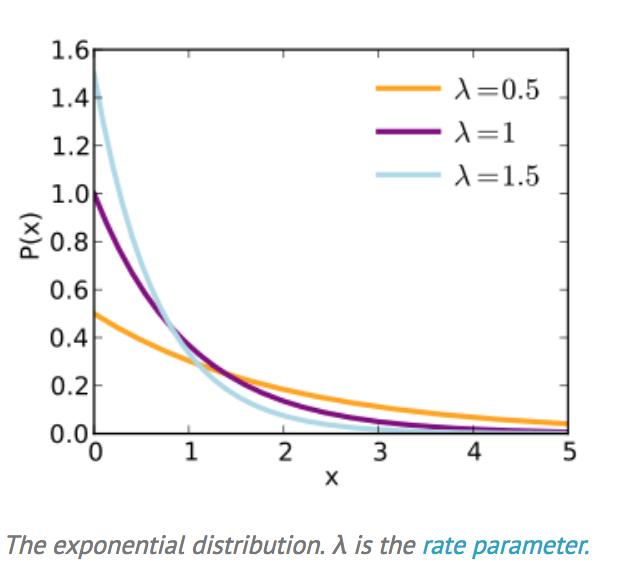
Let us imagine a situation in question 3, where you don’t put back the marble after you draw it. In that case, the distribution is [hypergeometric](https://en.wikipedia.org/wiki/Hypergeometric_distribution). If the number of marbles is largely relative to the number of draws, the distributions are similar because the chance of success changes less with each draw.

When people talk about picking marbles or balls from bags without replacement, it is Hypergeometric in nature. More broadly, it should come to mind when picking out a significant subset of a population as a sample.

To intuitively play around with Binomial distribution values: [Demonstration](http://demonstrations.wolfram.com/BinomialProbabilityDistribution/)

To read in detail on Binomial distribution: [Wolfram Binomial Distribution](http://mathworld.wolfram.com/BinomialDistribution.html)

**Exponential**



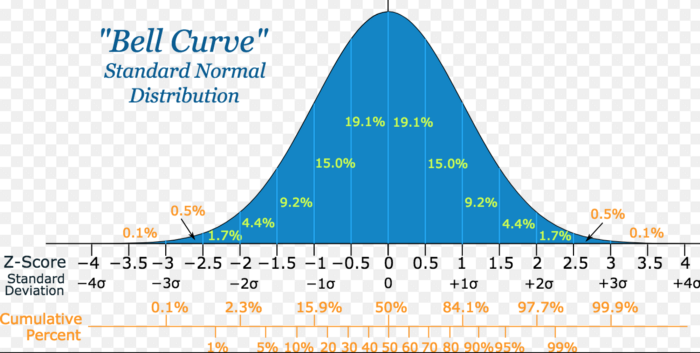
Source: <https://en.wikipedia.org/wiki/Exponential_distribution>

There is a strong relationship between the [Poisson distribution](https://www.statisticshowto.datasciencecentral.com/poisson-distribution/) and the Exponential distribution. For example, let’s say a Poisson distribution models the number of file requests on a server in a day. The time in between each file request can be modeled with an exponential distribution. The exponential distribution is mostly used for testing [product reliability](http://www.ni.com/white-paper/14412/en/). It’s also an important distribution for building continuous-time [Markov chains](https://en.wikipedia.org/wiki/Markov_chain).

Poisson’s “How many events per time?” in an experiment relates to the exponential’s “How long until an event?”.

Usually in continuous-time related questions/problems when we think about ‘time until event’, an exponential distribution can be used to model it.

**Normal and related distributions**



Source: <https://www.mathsisfun.com/data/standard-normal-distribution.html>

Save the best for the last. This is perhaps the most important of all. Its bell shape is instantly recognizable. Lets take an example of the Heights of people. If we plot that data, we will see that the bulk of people will be in the middle of that graph. If you take a bunch of values following the same distribution — any distribution — and sum them, the distribution of their sum follows (approximately) the normal distribution. The more things that are summed, the more their sum’s distribution matches the normal distribution. The fact that this is true regardless of the underlying distribution is amazing.

Probably, there could be something asked in the interview which relates to [central limit theorem](https://en.wikipedia.org/wiki/Central_limit_theorem) which is the concept central to probability distributions and relates to normal distribution.

Important Properties of a Normal distribution:

* The [mean, mode and median](https://www.purplemath.com/modules/meanmode.htm) are all equal.
* The curve is symmetric at the center (i.e. around the mean, μ).
* Exactly half of the values are to the left of center and exactly half the values are to the right.
* The total area under the curve is 1.

A [log-normal](https://en.wikipedia.org/wiki/Log-normal_distribution) (log-normal or Galton) distribution is a probability distribution with a normally distributed logarithm. If sums of things are normally distributed, then the products of things are log-normally distributed.

Let’s say you have a random sample taken from a normal distribution. The chi square distribution is the distribution of the sum of these random samples squared. It is based on the [chi-squared test](https://en.wikipedia.org/wiki/Pearson%27s_chi-squared_test) which is itself based on the sum of squares of differences, which are supposed to be normally distributed.

## VaR, ES.

23.4

## Arithmetic returns vs log-returns.

<https://python.plainenglish.io/arithmetic-vs-log-stock-returns-in-python-7f7c3cff125>

## What returns to use where.

<https://www.investopedia.com/terms/r/return.asp>

## Taylor series

<https://en.wikipedia.org/wiki/Taylor_series>

**Interview 2:**

## stochastic volatility,

23.3

## volatility smile,

23.3

## the Greeks,

1.14

## delta hedging for calls & puts

1.9

[](https://www.glassdoor.com/Interview/Introduce-abt-implied-vol-QTN_5322713.htm)

## Introduce abt implied vol

???

[](https://www.glassdoor.com/Interview/Expected-number-of-coin-flips-to-get-2-heads-in-a-row-QTN_4556728.htm)

## Expected number of coin flips to get 2 heads in a row

6:

<https://www.cs.cornell.edu/~ginsparg/physics/INFO295/mh.pdf>

[](https://www.glassdoor.com/Interview/X-Y-normal-random-variables-what-s-probability-of-2-x-y-1-what-s-probability-of-3-x-y-what-about-with-condition-x-QTN_4556729.htm)

## X, Y normal random variables, what's probability of 2\*x + y < 1, what's probability of 3\*x < y? what about with condition x > 0?

<https://stats.stackexchange.com/questions/144955/x-y-are-iid-from-n0-1-whats-the-probability-that-x2y>

## Can you produce a forward FX model for PLN/USD.

[**https://osf.io/ez6an/download**](https://osf.io/ez6an/download)

**Interview**

## two rounds interview, specifically mention the questions about VaR, and things like financial products. And also asking some behavioral questions relates to team management experiences and projects did in school

**Interview Questions**

* VaR, Option, CDS, swap, swaptions

# **Questions related to risk measures**

## How would you calculate Value at Risk (VaR)?

There are various methods, which produce VaR measures. We can distinguish four different basic methods of calculating VaR:

* **Variance Covariance approach,**
* **Historical Simulation approach,**
* **Monte Carlo Simulation and**
* **Extreme Value Theory.**

This chapter attempts to describe theoretically the methods, which we use in our research, and presents a brief summary of their advantages and disadvantages. Additionally, we present Exponentially Weighted Moving Average (EWMA), our methodology of forecasting variance.

According to the **variance covariance** approach all **market risks are normal and the portfolio is a linear function of these normal risks**. If normality holds the **VaR is the multiple of portfolio's standard deviation and the portfolio standard deviation is a linear function of individual volatilities and covariances**. In order to calculate VaR one needs to find the portfolio's standard deviation (using the variance covariance matrix and the weights of individual assets) and then multiply it by the value of the portfolio and the desired confidence level.

The formula for VaR is: VaR = -κ(α)\*Ρ\*σΡ (1)

Where σp is the portfolio's standard deviation, Ρ is the value of the portfolio and κ(α) is the desirable level of confidence (the (l-α)% quantile of the standard normal distribution).

Before someone can use this approach to calculate VaR should consider some other issues, which concern the nature of returns. As in the case of bonds (and generally fixed income instruments) and derivatives, returns are not linear functions of risk factors. One solution to this problem is the **delta-normal approach,** which takes a **first order approximation to the returns and then uses this approximation to calculate VaR**. **Delta-normal approach works by replacing the true positions with linear approximations and handling them in the same way as other linear positions.** Hence, we are assuming that the non-linearity in our position is sufficiently limited so that we can ignore it and still produce accurate VaR estimates**. This approach has some attractive aspects. The first is that maintains the linearity of the portfolio without adding any new risk factors. It also requires few additional data. By Delta-normal approach one can handle more complex positions without losing the benefits and convenience of the normal distribution.**

**However, one may need more precise VaR estimates without losing the convenience of working with the variance covariance method. This is the case of the second order or delta-gamma approach.** Second order approximation assumes that returns are normally distributed but allows for a non-linear relationship between the portfolio's value and the underlying returns. According to this method VaR can be calculated in two steps. The first step is the calculation of the first four moments (mean, standard deviation, skewness and kurtosis) of the portfolio's return distribution. The second step is to find a distribution that has the same four moments as the ones from portfolio's return distribution and then to find the a% quantile. After these steps the calculation of VaR is brought back to equation (1). Generally, for large portfolios where optionality is not the most important issue the delta normal approach is proper because it provides a fast and efficient calculation of VaR. On the other hand, for portfolios that are exposed to a small number of risks and with significant option components the delta gamma approach offers increased precision at a low computational cost. Variance covariance approach has many attractive features. First, this method, because of normality, makes the calculation of VaR very simple. Although the calculations are easy to implement, the figures of VaR produced by 65 this approach are very informative both on confidence level and holding period. Besides these, variance covariance method is very informative about the expected tail losses.

However, there are some drawbacks in this approach. **Its main problem is the excess kurtosis, which means that the returns distribution may have fat tails.** This derives from the assumption that normality holds and means that VaR will underestimate the expected losses. Moreover, the normal distribution assumes the symmetry of the returns distribution while in reality financial returns often exhibit asymmetric behavior. This negative aspect represents the problem of negative skewness. This is because the right tail of the distribution contains more data than the left one.

**Historical Simulation Approach** tries to find an empirical distribution of the rates of return assuming that past history carries out into the future. This method **uses the historical distribution of returns of a portfolio to simulate the portfolio's VaR. Often historical simulation is called non-parametric approach, because parameters like variances and covariances do not have to be estimated, as they are implicit in the data**. It is simple to implement if daily historical daily data have been collected. The choice of sample period influences the accuracy of VaR estimates. Longer periods provide better VaR estimates than short.

The first step on implementing this method involves identifying the instruments in a portfolio and collecting a sample of their historical returns. The second step is to calculate the simulated price of every instrument using the weights of the current portfolio (in order to simulate the returns in the next period). The third step assumes that the historical distribution that the returns follow is a good proxy for the returns in the next period. Historical Simulation uses the actual percentiles of the observation period as VaR measures. For instance for an observation period of 1000 days, the 95th percentile historical simulation VaR measure is the 51st largest logs observed in the sample of 1000 outcomes. That is because the five percent of the sample that should exceed the risk measure is equal to fifty losses**.**

**One of the greatest advantages of this method is that it does not depend on assumptions about the distribution of returns.** Therefore, the mistakes of assuming parametric distributions with thin tails where in reality the distributions of returns have fat tails are avoided. **One more positive characteristic is that the data set reflects gamma, vega risks as well as the correlations and volatilities.** Thus, there is no need for any parameter estimation. In relation to the previous characteristic Historical Simulation gives information about other useful statistics such as skewness and kurtosis. This approach does not make any distinction between the type of position and market risk (that is there are not different models for equities, bonds and derivatives like in variance covariance approach).

Although Historical Simulation seems to have many attractive characteristics, **there are some disadvantages. First, Historical Simulation results are dependent on the data set from the past, which may be too volatile or not, to predict the future**. Hence, one cannot provide accurate estimates of VaR because what happened in past will not necessarily happen in the future**. The same occurs when the period used for the estimation of VaR includes serious incidents (for example economic shocks), which are unlikely to happen in the future. This can also be reversed by assuming that important events can happen in future but because the data set (the observation period) does not include them the HS underestimates VaR. Second, Historical Simulation assumes that returns are independent and identically distributed. The data display time varying property of volatility.** For example, by choosing a long time series there is a problem in VaR estimation because too much emphasis is placed on data from the distant past. With short time series the estimations will probably not be reliable (because of the small length of the observation period**). Another drawback for this method is that it uses the same weights on all past observations.** If an observation from the distant past is excluded the VaR estimates may change significantly.

Forecasts are of great importance and widely used in economics and finance. It is reasonable that good forecasts lead to good decisions. The fact that volatility seems to cluster over time in a predictable way has very important implications for investors, risk managers and financial institutions. VaR increases as the volatility increases and vice-versa. There are many approaches for forecasting VaR. **This paper uses the exponentially weighted moving average to produce daily VaR forecasts. The EWMA approach for forecasting volatilities emphasizes on recent observations by using exponentially weighted moving averages of squared deviations**. **This method attaches different weights to the past observations contained in the observation period. Since the weights decline exponentially, the most recent receive much more weight than the earlier ones**. EWMA model calculates the volatility forecast for the next period as a weighted average of the previous period's volatility forecast and the current squared return. The formula for the EWMA model is:

where σt-1 represents the volatility of the returns at time t-1, rt-1 represents the return at time t-1 and λ is the decay factor. This method emphasizes that the volatility on a given day t-1 is actually used as a predictor for the volatility of the next day t. The EWMA model depends on the decay factor. The parameter λ (01) determines the relative weights that are applied to the returns and the effective amount of data that are used in volatility estimation. In this paper the decay factor is set to be 0.94 just like Risk Metrics for daily data2 . This method is equivalent to the Integrated GARCH or IGARCH (1,1) family of popular conditional models. It can be viewed as a special case of GARCH process, which is given by the equation: where α0 , α1 and α2 are parameters. By setting α0 to 0 and α1 and α2 sum to unity the GARCH model becomes an exponentially weighted estimator. EWMA approach explains volatility clustering, which means that abnormal volatility in one period is likely to lead to abnormal volatility in the next period. It also allows the volatility to vary from one period to another. A very important feature of the exponentially weighted estimator is that it can be written in recursive form in order to be used for making volatility forecasts. To derive this form one should assume that an infinite amount of data is available. This recursive form can be written as follows…

## What's wrong with VaR as a measurement of risk?

It is not a sub-additive measure.

It does not say anything about losses beyond the confidence level.

## What is non-Linear VaR? How would you calculate it?

**Nonlinear Considerations**

Nonlinear risk exposure arises in the VaR calculation of a portfolio of nonlinear derivatives. Nonlinear derivatives, such as options, depend on a variety of characteristics, including implied volatility, time to maturity, underlying asset price, and the current interest rate.

It is difficult to collect the historical data on the returns because the option returns would need to be conditioned on all of the characteristics to use the standard VaR approach. Inputting all of the characteristics associated with options into the Black-Scholes model or another option pricing model causes the models to be nonlinear due to the nature of the derivative. Therefore, the payoff curves are nonlinear because the corresponding value is not proportional to the input due to the time and volatility portion of the model, in particular since options are wasting assets.

The nonlinearity of certain derivatives leads to nonlinear risk exposures in the VaR of a portfolio. Nonlinearity can be witnessed in the payoff diagram of a plain vanilla call option. The payoff diagram has a strong positive convex payoff profile before the option's expiration date, with respect to the stock price.

When the call option reaches a point where the option is in the money, it reaches a point where the payoff becomes linear. Conversely, as a call option becomes increasingly out of the money, the rate at which the option loses money decreases until the option premium is zero.

**Kurtosis**

If a portfolio includes nonlinear derivatives, such as options, the distribution of the portfolio returns will have a positive or negative skew or high or low kurtosis. The skewness measures the asymmetry of a probability distribution around its mean. Kurtosis measures the distribution around the mean; a high kurtosis has fatter tail ends of the distribution, and a low kurtosis has skinny tail ends of the distribution.

Therefore, it is difficult to use the VaR method that assumes the returns are normally distributed. Instead, the VaR calculation of a portfolio containing nonlinear exposures is usually calculated using Monte Carlo VaR simulations of options pricing models to estimate the VaR of the portfolio.

**4. What is the parametric method of calculating VaR? What are its advantages?**

The nonparametric method does not require that the population being analyzed meet certain assumptions, or parameters. This gives analysts a great deal of flexibility and allows for qualitative or ordinal variables to be included. Although nonparametric statistics have the advantage of having to meet few assumptions, they are less powerful than parametric statistics. This means that they may not show a relationship between two variables when in fact one exists. As a result, most risk managers prefer a more quantitative approach.

The parametric method, also known as the variance-covariance method, is a risk management technique for calculating the VaR of a portfolio of assets that first identifies the mean, or expected value, and standard deviation of an investment portfolio. The parametric method looks at the price movements of investments over a look-back period and uses probability theory to compute a portfolio's maximum loss. The variance-covariance method for the value at risk calculates the standard deviation of price movements of an investment or security. Assuming stock price returns and volatility follow a normal distribution, the maximum loss within the specified confidence level is calculated.

**Example with One Security**

Consider a portfolio that includes only one security, stock ABC. Suppose $500,000 is invested in stock ABC. The standard deviation over 252 days, or one trading year, of stock ABC, is 7%. Following the normal distribution, the one-sided 95% confidence level has a z-score of 1.645.

The value at risk in this portfolio is

*$57,575 = ($500,000\*1.645\*.07).*

Therefore, with 95% confidence, the maximum loss will not exceed $57,575 in a given trading year.

**Example with Two Securities**

The value at risk of a portfolio with two securities can be determined by first calculating the portfolio's volatility. Multiply the square of the first asset's weight by the square of the first asset's standard deviation and add it to the square of the second asset's weight multiplied by the square of the second asset's standard deviation. Add that value to two multiplied by the weights of the first and second assets, the correlation coefficient between the two assets, asset one's standard deviation, and asset two's standard deviation. Then multiply the square root of that value by the z-score and the portfolio value.

For example, suppose a risk manager wants to calculate the value at risk using the parametric method for a one-day time horizon. The weight of the first asset is 40%, and the weight of the second asset is 60%. The standard deviation is 4% for the first and 7% for the second asset. The correlation coefficient between the two is 25%. The z-score is -1.645. The portfolio value is $50 million.

The parametric value at risk over a one-day period, with a 95% confidence level, is:

*$3.99 million = ($50,000,000\*-1.645)\*√(0.4^2\*0.04^2)+(0.6^2\*0.07^2)+[2(0.4\*0.6\*0.25\*0.04\*0.07\*)]*

**The Bottom Line**

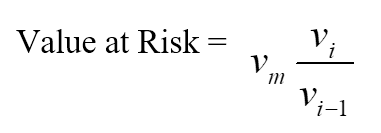
If a portfolio has multiple assets, its volatility is calculated using a matrix. A variance-covariance matrix is computed for all the assets. The vector of the weights of the assets in the portfolio is multiplied by the transpose of the vector of the weights of the assets multiplied by the covariance matrix of all of the assets.

In practice, the calculations for VaR are typically done through financial models. Modeling functions will vary depending on whether the VaR is being calculated for one security, two securities, or a portfolio with three or more securities.

**5. What is the historical method of calculating VaR? What are its advantages?**

The historical method is the **simplest** method for calculating Value at Risk. Market data for the last 250 days is taken to calculate the percentage change for each risk factor on each day. Each percentage change is then calculated with current market values to present 250 scenarios for future value.

For each of the scenarios, the portfolio is valued using full, non-linear pricing models. The third worst day selected is assumed to be 99% VaR.



Where:

* vi is the number of variables on day i
* m is the number of days from which historical data is taken

**6. Why would you calculate VaR using Monte Carlo simulations?**

Under the Monte Carlo method, Value at Risk is calculated by randomly creating a number of scenarios for future rates using non-linear pricing models to estimate the change in value for each scenario, and then calculating the VaR according to the worst losses.

The Monte Carlo method is suitable for a great range of risk measurement problems, especially when dealing with complicated factors. It assumes that there is a known probability distribution for risk factors.

**7. What are the challenges in calculating VaR for a mixed portfolio?**

Need to measure not only return and volatility of individual assets, but also the correlations between them. When the number and diversity of positions grow, the difficulty and cost of measuring risk grows exponentially.

The VAR of A&B, is NOT the sum of VAR A and VAR B. Need to factor in correlation, sometimes a higher individual VAR can result in a lower portfolio VAR.

**8. What's GVAR? How can you calculate it?**

GVAR stands for Global Vector Autoregression. It is a two step process where individual VAR models are constructed for each economy. The Individual VAR's are then stacked and solved simultaneously. Some econometric packages have GVAR capabilities included.

**9. What is the one-day VaR of a $50m portfolio with a daily standard deviation of 2% at a 95% confidence level? What is the annualized VaR?**

[**https://www.academia.edu/23706844/Bank\_Risk\_Management\_Lecture\_4\_Value\_at\_Risk\_VaR**](https://www.academia.edu/23706844/Bank_Risk_Management_Lecture_4_Value_at_Risk_VaR)

*1-day VaR = -2% \* 1.64 \* 50 MM*

*1-day VaR = -2% \* 1.64 \* 50 MM \*sqrt(252)*

**10. What do you know about extreme value theory?**

Extreme value theory or extreme value analysis (EVA) is a branch of statistics dealing with the extreme deviations from the median of probability distributions. It seeks to assess, from a given ordered sample of a given random variable, the probability of events that are more extreme than any previously observed. Extreme value analysis is widely used in many disciplines, such as structural engineering, finance, earth sciences, traffic prediction, and geological engineering. For example, EVA might be used in the field of hydrology to estimate the probability of an unusually large flooding event, such as the 100-year flood. Similarly, for the design of a breakwater, a coastal engineer would seek to estimate the 50-year wave and design the structure accordingly.

[**https://en.wikipedia.org/wiki/Extreme\_value\_theory**](https://en.wikipedia.org/wiki/Extreme_value_theory)

**11. What is Expected Shortfall? How is it calculated? Why is it considered better than VaR? What are the disadvantages?**

Conditional Value at Risk (CVaR), also known as the expected shortfall, is a risk assessment measure that quantifies the amount of tail risk an investment portfolio has. CVaR is derived by taking a weighted average of the “extreme” losses in the tail of the distribution of possible returns, beyond the value at risk (VaR) cutoff point. Conditional value at risk is used in portfolio optimization for effective risk management.

**Understanding Conditional Value at Risk (CVaR)**

Generally speaking, if an investment has shown stability over time, then the value at risk may be sufficient for risk management in a portfolio containing that investment. However, the less stable the investment, the greater the chance that VaR will not give a full picture of the risks, as it is indifferent to anything beyond its own threshold.

Conditional Value at Risk (CVaR) attempts to address the shortcomings of the VaR model, which is a statistical technique used to measure the level of financial risk within a firm or an investment portfolio over a specific time frame. While VaR represents a worst-case loss associated with a probability and a time horizon, CVaR is the expected loss if that worst-case threshold is ever crossed. CVaR, in other words, quantifies the expected losses that occur beyond the VaR breakpoint.

**Conditional Value at Risk (CVaR) Formula**

Since CVaR values are derived from the calculation of VaR itself, the assumptions that VaR is based on, such as the shape of the distribution of returns, the cut-off level used, the periodicity of the data, and the assumptions about stochastic volatility, will all affect the value of CVaR. Calculating CVaR is simple once VaR has been calculated. It is the average of the values that fall beyond the VaR

**​Conditional Value at Risk and Investment Profiles**

Safer investments like large-cap U.S. stocks or investment-grade bonds rarely exceed VaR by a significant amount. More volatile asset classes, like small-cap U.S. stocks, emerging markets stocks, or derivatives, can exhibit CVaRs many times greater than VaRs. Ideally, investors are looking for small CVaRs. However, investments with the most upside potential often have large CVaRs.

Financially engineered investments often lean heavily on VaR because it doesn't get bogged down in outlier data in models. However, there have been times where engineered products or models may have been better constructed and more cautiously used if CVaR had been favored. History has many examples, such as Long-Term Capital Management which depended on VaR to measure its risk profile, yet still managed to crush itself by not properly taking into account a loss larger than forecasted by the VaR model. CVaR would, in this case, have focused the hedge fund on the true risk exposure rather than the VaR cutoff. In financial modeling, a debate is almost always going on about VaR versus CVaR for efficient risk management.

**12. What are the strengths and weaknesses of historical simulation, Monte-Carlo simulation, and Variance-Covariance method in VaR calculation?**

* If We look at past crises we can see how a particular portfolio would have fared in these crises. Major drawback. History seldom repeats itself (but it does rhyme)
* With Monte Carlo we do a large number of simulated tress tests but you have to have a decent knolwledge of the distributions of the stressors and their linkages.
* VCOV allows for inter-relationships in the returns however these covariances are usually not stable and Long term covariarances are typically smaller than short term covariances

**13. What is expected shortfall?**

Expected shortfall is a risk measure sensitive to the shape of the tail of the distribution of returns on a portfolio, unlike the more commonly used value-at-risk (VAR). Expected shortfall is calculated by averaging all of the returns in the distribution that are worse than the VAR of the portfolio at a given level of confidence. For instance, for a 95% confidence level, the expected shortfall is calculated by taking the average of returns in the worst 5% of cases.

**14. What is incremental default risk?**

**Definition**

The term Incremental Default Rate is used in the context of multi-period credit risk analysis to denote the empirical (or modelled) default rate observed in a certain portfolio during a defined sub-interval.

The incremental default rate can be considered as the building block of the [Cumulative Default Rate](https://www.openriskmanual.org/wiki/Cumulative_Default_Rate). Observing whether an entity is defaulted over a period [t_{k-1}, t_k], the incremental default rate is denoted \mbox{IDR}_{k}

**Formula**

The incremental default rate during period k, given an initial count of N0, and an incremental default count of NtD is given by


\mbox{IDR}_{t} = \frac{N^{D}_t}{N_0}


NB: The important difference of this formula compared to the regular [Default Rate](https://www.openriskmanual.org/wiki/Default_Rate) expression is that the denominator refers to initial count of the portfolio / cohort

# *****Questions on the yield curve:*****

**15. What are the uses of the yield curve?**

**What Is a Yield Curve?**

A yield curve is a line that plots yields (interest rates) of bonds having equal credit quality but differing maturity dates. The slope of the yield curve gives an idea of future interest rate changes and economic activity.

There are three main shapes of yield curve shapes: normal (upward sloping curve), inverted (downward sloping curve), and flat.

**KEY TAKEAWAYS**

Yield curves plot interest rates of bonds of equal credit and different maturities.

The three key types of yield curves include normal, inverted, and flat. Upward sloping (also known as normal yield curves) is where longer-term bonds have higher yields than short-term ones.

While normal curves point to economic expansion, downward sloping (inverted) curves point to economic recession.

Yield curve rates are published on the Treasury’s website each trading day.

**How a Yield Curve Works**

A yield curve is used as a benchmark for other debt in the market, such as mortgage rates or bank lending rates, and it is used to predict changes in economic output and growth. The most frequently reported yield curve compares the three-month, two-year, five-year, 10-year, and 30-year U.S. Treasury debt. Yield curve rates are usually available at the Treasury's interest rate websites by 6:00 p.m. ET each trading day.

A normal yield curve is one in which longer maturity bonds have a higher yield compared to shorter-term bonds due to the risks associated with time. An inverted yield curve is one in which the shorter-term yields are higher than the longer-term yields, which can be a sign of an upcoming recession. In a flat or humped yield curve, the shorter- and longer-term yields are very close to each other, which is also a predictor of an economic transition.

**Types of Yield Curves**

**Normal Yield Curve**

A normal or up-sloped yield curve indicates yields on longer-term bonds may continue to rise, responding to periods of economic expansion. A normal yield curve thus starts with low yields for shorter-maturity bonds and then increases for bonds with longer maturity, sloping upwards. This is the most common type of yield curve as longer-maturity bonds usually have a higher yield to maturity than shorter-term bonds.

For example, assume a two-year bond offers a yield of 1%, a five-year bond offers a yield of 1.8%, a 10-year bond offers a yield of 2.5%, a 15-year bond offers a yield of 3.0%, and a 20-year bond offers a yield of 3.5%. When these points are connected on a graph, they exhibit a shape of a normal yield curve.

A normal yield curve implies stable economic conditions and should prevail throughout a normal economic cycle. A steep yield curve implies strong economic growth in the future—conditions that are often accompanied by higher inflation, which can result in higher interest rates.

**Inverted Yield Curve**

An inverted yield curve instead slopes downward and means that short-term interest rates exceed long-term rates. Such a yield curve corresponds to periods of economic recession, where investors expect yields on longer-maturity bonds to become even lower in the future.

Moreover, in an economic downturn, investors seeking safe investments tend to purchase these longer-dated bonds over short-dated bonds, bidding up the price of longer bonds driving down their yield.

An inverted yield curve is rare but is strongly suggestive of a severe economic slowdown. Historically, the impact of an inverted yield curve has been to warn that a recession is coming.

**Flat Yield Curve**

A flat yield curve is defined by similar yields across all maturities. A few intermediate maturities may have slightly higher yields, which causes a slight hump to appear along the flat curve. These humps are usually for the mid-term maturities, six months to two years.

As the word flat suggests, there is little difference in yield to maturity among shorter and longer-term bonds. A two-year bond could offer a yield of 6%, a five-year bond 6.1%, a 10-year bond 6%, and a 20-year bond 6.05%.

Such a flat or humped yield curve implies an uncertain economic situation. It may come at the end of a high economic growth period that is leading to inflation and fears of a slowdown. It might appear at times when the central bank is expected to increase interest rates.

In times of high uncertainty, investors demand similar yields across all maturities.

**What Is a U.S. Treasury Yield Curve?**

The U.S. Treasury yield curve refers to a line chart that depicts the yields of short-term Treasury bills compared to the yields of long-term Treasury notes and bonds. The chart shows the relationship between the interest rates and the maturities of U.S. Treasury fixed-income securities. The Treasury yield curve (also referred to as the term structure of interest rates) shows yields at fixed maturities, such as one, two, three, and six months and one, two, three, five, seven, 10, 20, and 30 years. Because Treasury bills and bonds are resold daily on the secondary market, yields on the notes, bills, and bonds fluctuate.

**What Is Yield Curve Risk?**

Yield curve risk refers to the risk investors of fixed-income instruments (such as bonds) experience from an adverse shift in interest rates. Yield curve risk stems from the fact that bond prices and interest rates have an inverse relationship to one another. For example, the price of bonds will decrease when market interest rates increase. Conversely, when interest rates (or yields) decrease, bond prices increase.

**How Can Investors Use the Yield Curve?**

Investors can use the yield curve to make predictions on where the economy might be headed and use this information to make their investment decisions. If the bond yield curve indicates an economic slowdown might be on the horizon, investors might move their money into defensive assets that traditionally do well during recessionary times, such as consumer staples. If the yield curve becomes steep, this might be a sign of future inflation. In this scenario, investors might avoid long-term bonds with a yield that will erode against increased prices.

**16. What's the riskiest part of the yield curve?**

OK, good question. The answer - for you - actually depends on the type of investing you do. Because “risk” to you isn’t just on the volatility of the curve, it is also in the duration of the investments.

It looks like the other answers provided have a similar “take” on “riskiness”, but view the question a bit differently. So I’m trying for simplicity. If this isn’t clear, I can always try again for you.

“Risk” is often measured by the volatility (observed change) of the total returns (earnings plus price gains). Greater volatility means greater risk. The short end of the yield curve has traditionally been most likely to change suddenly and by large amounts based on market events or central bank actions. The long end of the YC is notorious for not moving when the central bank wants it to, but only when other market factors induce it to move. This is how an inverted yield curve can happen.

The duration of your investments is short if you are invested there. Not so the long end of the yield curve. A small change in the YC at the long end disproportionately hits long investments. This is basic bond math.

What is that “bond math”? In other words, a change of 1 basis point in rates has a 1 basis point impact on price for each year you have to hold it until your investment pays you back your invested funds on a PV basis (the duration). So the ten year investment has more price volatility for a given change in yields at that end of the curve. It takes ten times the movement in 1 year rates to have the same price impact on a short investment as the movement needed on a 10 year duration investment.

So you can take both rate risk and duration risk, depending on what you choose to hold.

**17. What does it mean for risk when the yield curve is inverted?**

* The yield curve is a line chart that plots interest rates for bonds that have equal credit quality, but different maturity dates.
* Yields are normally higher for bonds that mature over longer periods, as investors are rewards for holding bonds for more time.
* An inverted yield curve is when interest rates on long-term bonds fall lower than those of short-term bonds.
* This can be a sign of a coming recession – an inverted yield curve has emerged roughly a year before nearly all recessions since 1960.

**18. What is the discount factor? How would you calculate it?**

The general discount factor formula is: Discount Factor = 1 / (1 \* (1 + Discount Rate)Period Number) To use this formula, you'll need to find out the periodic interest rate or discount rate. This can easily be determined by dividing the annual discount factor interest rate by the total number of payments per year.

**19. What is convexity? How would you calculate it? Why is it important?**

**Convexity:**

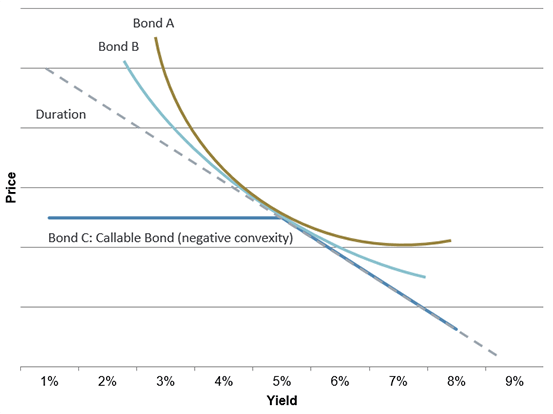
As the yield on a bond changes so too does its duration. A bond’s convexity measures **the sensitivity of a bond’s duration to changes in yield.** Duration is an imperfect way of measuring a bond’s price change, as it indicates that this change is linear in nature when in fact it exhibits a sloped or “convex” shape. A bond is said to have **positive convexity if duration rises as the yield declines.** A bond with positive convexity will have **larger price increases due to a decline in yields than price declines due to an increase in yields.** Positive convexity can be thought of as working in the **investor’s favor**, since the **price becomes less sensitive when yields rise** (**prices down)** than when **yields decline (prices up).** Bonds can also **have negative convexity**, which would indicate that **duration rises as yields increase and can work against an investor’s interest**. The table below highlights the types of bonds that exhibit each type of convexity.

**Examples of Bonds with Positive and Negative Convexity**

|  |  |
| --- | --- |
| Type of Convexity | Typical Types of Bonds |
| Positive Convexity | Non-callable bonds, bonds with make-whole calls |
| Negative Convexity | MBS (most), bonds with a traditional call, preferreds |

A useful way to visualize a bond’s convexity is to plot the potential price change against various yields. If two bonds have the same duration and yield but differing convexities, a change in interest rates will affect each bond differently. For example, the chart below shows three bonds: a bond with higher positive convexity (Bond A) will be less affected by interest rates than a bond with lower positive convexity (Bond B). On the other hand, a bond with negative convexity (Bond C) will exhibit larger price fluctuations should rates rise than if they were to fall.

**Bonds Can Have Very Different Convexities: Positive vs. Negative**



**20. What's the relationship between coupon rate and convexity?**

Typically, **the higher the coupon rate or yield, the lower the convexity**—or market risk—of a bond. This lessening of risk is because market rates would have to increase greatly to surpass the coupon on the bond, meaning there is less interest rate risk to the investor.

**21. What's the meaning of duration? Is it constant for all yields?**

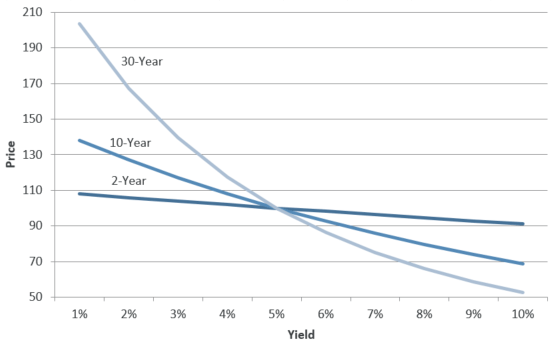
In simple terms, modified duration gives an idea of how the **price of a bond will be affected should interest rates change**. A higher duration implies greater price volatility should rates move. Duration is quoted as the percentage change in price for each given percent change in interest rates. For example, the price of a bond with a duration of 2 would be expected to increase (decline) by about 2.00% for each 1.00% move down (up) in rates.

The duration of a bond is primarily affected by its coupon rate, yield, and remaining time to maturity.

* **The duration of a bond will be higher the lower its coupon**.
* **Duration will be higher the lower its yield.**
* **Duration will also be higher the longer its maturity**.

The following scenarios of comparing two bonds should help clarify how these three traits affect a bond’s duration:

* **If the coupon and yield are the same, duration increases with time left to maturity**
* **If the maturity and yield are the same, duration increases with a lower coupon**
* **If the coupon and maturity are the same, duration increases with a lower yield**



**Example:**

|  |  |  |  |
| --- | --- | --- | --- |
| **If Rates Move Up ...** | **2-Year Bond** | **10-Year Bond** | **30-Year Bond** |
| 1.00% | -1.0% | -6.9% | -13.7% |
| 2.00% | -1.9% | -13.2% | -24.7% |
| 3.00% | -2.8% | -19.0% | -33.6% |

# ****Questions on quantitative concepts:****

**25. Can you explain the assumptions behind Black Scholes?**

- No dividends

- Stock returns follow Normal distribution, with constant volatility

- No asymmetries of information

- No arbitrage opportunities (markets are random, not predictable)

- No transactions costs

- Fixed risk-free rate

- Complete / liquid markets

- The option is European and can only be exercised at expiration

**26. What's a volatility smile? Why does it occur? What are the implications for Black Scholes?**

The observations that volatilities of options with different strikes implied from market prices are not equivalent to BS model values but, instead, they are heavily dependent on the strike value. They often create “a smile”, i.e. the impl vol is higher in case of strikes lower than initial stock price (often explained by market fear) and sometimes also higher in case of much higher strikes (market greed).

This observation violates the BS model assumptions (const vol). It can be relaxed through an introduction of stochastic volatility in the model (e.g. Heston model).

**27. What are the Greeks?**

The derivative of option prices presenting the option sensitivities wrt to BS model parameters.

Examples:

- Delta – wrt stock price

- Vega – wrt volatility

- pho - wrt risk-free rate

- Theta – wrt time

- Gamma – second deriv wrt Stock

- Higher order derivatives

**28. How are the main Greeks derived?**

<https://bookdown.org/maxime_debellefroid/MyBook/the-greeks.html>

<https://www.youtube.com/watch?v=lWWW3xlBHpY>

**29. What do you know about jump processes?**

A jump process is a type of stochastic process that has discrete movements, called jumps, with random arrival times, rather than continuous movement, typically modelled as a simple or compound Poisson process.

In finance, various stochastic models are used to model the price movements of financial instruments; for example the Black–Scholes model for pricing options assumes that the underlying instrument follows a traditional diffusion process, with continuous, random movements at all scales, no matter how small. John Carrington Cox and Stephen Ross  proposed that prices actually follow a 'jump process'.

Robert C. Merton extended this approach to a hybrid model known as jump diffusion, which states that the prices have large jumps interspersed with small continuous movements.

* Poisson process, an example of a jump process
* Continuous-time Markov chain (CTMC), an example of a jump process and a generalization of the Poisson process
* Counting process, an example of a jump process and a generalization of the Poisson process in a different direction than that of CTMCs
* Interacting particle system, an example of a jump process
* Kolmogorov equations (continuous-time Markov chains)

Merton (1976) and Cox and Ross (1976) were the first to allow the stock to jump ‘up’ or ‘down’, engendering a discontinuity in the stock price process. Using adequate parameters, Merton’s model was able to generate a lots of volatility smiles and skews. Particularly, choosing a negative mean for the jump process can readily capture short-term skews. Simultaneously, the model retains the undesirable independence property. Numerous studies on jump diffusion models have been undertaken since that time.

**30. Should you use implied standard deviation or historical deviation to forecast volatility? Explain your choice.**

**Short Answer**

Volatility is annualized standard deviation of returns. Or is it? Because that is a statistical measure, necessarily backward looking, and because volatility seems to vary, and we want to know what it will be in the future, and because people have different views on what volatility will be in the future, things are not that simple.

**Example**

Actual volatility is the *σ* that goes into the Black–Scholes partial differential equation. Implied volatility is the number in the Black–Scholes formula that makes a theoretical price match a market price.

**Long Answer**

**Actual volatility** is a measure of the amount of randomness in a financial quantity at any point in time. It’s what Desmond Fitzgerald calls the ‘bouncy, bouncy.’ It’s difficult to measure, and even harder to forecast but it’s one of the main inputs into option pricing models. It’s difficult to measure since it is defined mathematically via standard deviations which requires historical data to calculate. Yet actual volatility is not a historical quantity but an instantaneous one. Realized/historical volatilities are associated with a period of time, actually two periods of time. We might say that the daily volatility over the last sixty days has been 27%. This means that we take the last sixty days’ worth of daily asset prices and calculate the volatility. Let me stress that this has two associated timescales, whereas actual volatility has none. This tends to be the default estimate of future volatility in the absence of any more sophisticated model. For example, we might assume that the volatility of the next sixty days is the same as over the previous sixty days. This will give us an idea of what a sixty-day option might be worth. Implied volatility is the number you have to put in to the Black–Scholes option-pricing equation to get the theoretical price to match the market price. Often said to be the market’s estimate of volatility.

Let’s recap. We have **actual volatility** which is the instantaneous amount of noise in a stock price return. It is sometimes modelled as a simple constant, sometimes as time dependent, sometimes as stock and time dependent, sometimes as stochastic and sometimes as a jump process, and sometimes as uncertain, that is, lying within a range. It is impossible to measure exactly, the best you can do is to get a statistical estimate based on past data. But this is the parameter we would dearly love to know because of its importance in pricing derivatives. Some hedge funds believe that their edge is in forecasting this parameter better than other people, and so profit from options that are mispriced in the market.

Since you can’t see actual volatility people often rely on measuring **historical** or **realized volatility**. This is a backward looking statistical measure of what volatility has been. And then one assumes that there is some information in this data that will tell us what volatility will be in the future. There are several models for measuring and forecasting volatility and we will come back to them shortly.

**Implied volatility** is the number you have to put into the Black–Scholes option-pricing *formula* to get the theoretical price to match the market price. This is often said to be the market’s estimate of volatility. More correctly, option prices are governed by supply and demand. Is that the same as the market taking a view on future volatility? Not necessarily because most people buying options are taking a directional view on the market and so supply and demand reflects direction rather than volatility. But because people who hedge options are not exposed to direction only volatility it looks to them as if people are taking a view on volatility when they are more probably taking a view on direction, or simply buying out-of-the-money puts as insurance against a crash. For example, the market falls, people panic, they buy puts, the price of puts and hence implied volatility goes up. Where the prices tops depends on supply and demand, not on anyone’s estimate of future volatility, within reason. Implied volatility levels the playing field so you can compare and contrast option prices across strikes and expirations.

There is also **forward volatility**. The adjective ‘forward’ is added to anything financial to mean values in the future. So forward volatility would usually mean volatility, either actual or implied, over some time period in the future.

Finally **hedging volatility** means the parameter that you plug into a delta calculation to tell you how many of the underlying to sell short for hedging purposes. Since volatility is so difficult to pin down it is a natural quantity for some interesting modelling. Here are some of the approaches used to model or forecast volatility.

*Econometric models:* These models use various forms of time series analysis to estimate current and future expected actual volatility. They are typically based on some regression of volatility against past returns and they may involve autoregressive or moving-average components. In this category are the **GARCH** type of models. Sometimes one models the square of volatility, the and not just closing prices, and sometimes one models the logarithm of volatility. The latter seems to be quite promising because there is evidence that actual volatility is log normally distributed. Other work in this area decomposes the volatility of a stock into components, market volatility, industry volatility and firm-specific volatility. This is similar to **CAPM** for returns.

*Deterministic models:* The simple Black–Scholes formulae assume that volatility is constant or time dependent. But market data suggests that implied volatility varies with strike price. Such market behaviour cannot be consistent with a volatility that is a deterministic function of time. One way in which the Black–Scholes world can be modified to accommodate strike-dependent implied volatility is to assume that actual volatility is a function of both time and the price of the underlying. This is the **deterministic volatility** (**surface**) model. This is the simplest extension to the Black–Scholes world that can be made to be consistent with market prices. All it requires is that we have *σ*(*S*, *t*), and the Black–Scholes partial differential equation is still valid. The interpretation of an option’s value as the present value of the expected payoff under a risk-neutral random walk also carries over. Unfortunately the Black–Scholes closed form formulć are no longer correct. This is a simple and popular model, but it does not capture the dynamics of implied volatility very well.

*Stochastic volatility:* Since volatility is difficult to measure , and seems to be forever changing, it is natural to model it as stochastic. The most popular model of this type is due to Heston. Such models often have several parameters which can either be chosen to fit historical data or, more commonly, chosen so that theoretical prices calibrate to the market. Stochastic volatility models are better at capturing the dynamics of traded option prices better than deterministic models. However, different markets behave differently. Part of this is because of the way traders look at option prices. Equity traders look at implied volatility versus strike, FX traders look at implied volatility versus delta. It is therefore natural for implied volatility curves to behave differently in these two markets. Because of this there have grownup the sticky strike, sticky delta, etc., models, which model how the implied volatility curve changes as the underlying moves.

*Poisson processes:* There are times of low volatility and times of high volatility. This can be modelled by volatility that jumps according to a Poisson process.

*Uncertain volatility:* An elegant solution to the problem of modelling the unseen volatility is to treat it as uncertain, meaning that it is allowed to lie in a specified range but whereabouts in that range it actually is, or indeed the probability of being at any value, are left unspecified. With this type of model we no longer get a single option price, but a range of prices, representing worst-case

scenario and best-case scenario.

# ****Questions on hedging:****

**31. What is delta hedging?**

**Delta hedging** One of the building blocks of derivatives theory is delta hedging. This is the theoretically perfect elimination of all risk by using a very clever hedge between the option and its underlying. Delta hedging exploits the perfect correlation between the changes in the option value and the changes in the stock price. This is an example of ‘dynamic’ hedging; the hedge must be continually monitored and frequently adjusted by the sale or purchase of the underlying asset. Because of the frequent rehedging, any dynamic hedging strategy is going to result in losses due to transaction costs. In some markets this can be very important. The ‘underlying’ in a delta-hedged portfolio could be a traded asset, a stock for example, or it could be another random quantity that determines a price such as a risk of default. If you have two instruments depending on the same risk of default, you can calculate the sensitivities, the deltas, of their prices to this quantity and then buy the two instruments in amounts inversely proportional to these deltas (one long, one short). This is also delta hedging. If two underlyings are very highly correlated you can use one as a proxy for the other for hedging purposes. You would then only be exposed to basis risk. Be careful with this because there may be times when the close relationship breaks down. If you have many financial instruments that are uncorrelated with each other then you can construct a portfolio with much less risk than any one of the instruments individually. With a large such portfolio you can theoretically reduce risk to negligible levels. Although this

**32. How would you hedge against a particular equity/bond under current market conditions?**

You can hedge by **buying put options at lower levels** to protect your stock price. You can also hedge by **selling futures against your cash market holding**. If you are holding bonds, you can hedge by **keeping duration the same as your time frame**. This saves you from any interest risk. Also, **avoid risky bonds.**

**33. When can hedging an options position mean that you take on more risk?**

1. Hedging an options position can become riskier if one side of the option position is left open. Also if you hedge through covered calls, it can be risky in the event of a fall in the stock price. Normally, volatility is positive for options.

2. Hedging can increase your risk if you are forced to both buy short-dated options and hedge them.

3. E.g. on Monday you get forced to buy some Friday expiry OTM puts, say 95% strike S&P weeklies. Of course, you go and buy some delta against them to "hedge" yourself. Next thing you know, the the market tanks. Unfortunately, by Friday it's only down 3.5%, so it's does not fall far enough to reach the strike. So, on Friday expiration, you are out your premium and down money on your delta.

Overall, it's a pretty typical "painful moment" for a market maker. Usually, in this case you are better off selling some ATMish gamma and leaving the tinys to decay in peace. Should the market really take a dive, you got some lottery tickets.

4. Hedging does not increase your risk in this particular example: You take on delta exposure by buying the short dated option outright. Thus buying/selling underlying (put/call) in any case will reduce your delta exposure, hence risk of changes in the underlying, given you hedge the right amount and at the right timing (this is venture is impossible to generalize as it applies differently to each case). Now, you are long gamma but being long gamma does not guarantee at all that you end up better off not hedging initially. If your boss instructs you to be at all times almost perfectly delta hedged (most French bosses are anal about this, probably because they are horrible delta traders) then you hedge, period. It reduces your delta exposure, hence risk in moves in the underlying. It is complete nonsense to start arguing in retrospect that no hedge may have resulted in a better payoff because the underlying followed a price path not anticipated earlier.

I concur with Strange that there are often better ways to hedge than always going through the underlying but I disagree with him that it poses a "painful moment" to market markers. Market makers who are dependent on the market moving in specific ways are probably very bad volatility traders. Your job as market maker is to earn money from the bid/offer spread and to reduce your risk exposure to lower moment greeks, given it is feasible and cost-efficient. The other times you, as market maker, attempt to benefit from what you perceive as mispricings in the option valuation. Thus, hedging the long options position with the underlying reduces your risk, period. There are obviously exceptions to this, for example, when the underlying is so extremely illiquid that it would be prohibitive to hedge/re-hedge frequently. But it has to really be analyzed in context. But if the interview book looked for a straight forward answer which applies to most cases then hedging reduces your risk, simple as that.

**34. An option is at the money. How many shares of stock should you hold to hedge it?**

The delta of an option tells you the relationship between the underlying security's price and the option's price itself.”

The delta represents the amount the price of the derivative will change when there is a change in the underlying price. For example, suppose you buy a call option with a delta of 0.2. The call option will increase by 50 cents for every $1 increase in the price of the underlying asset.

The delta of an option helps you determine the quantity of the underlying asset to buy or sell. This is known as delta hedging. Delta hedging involves trading another security to create a delta-neutral position, or a position that has a zero delta.

You can use delta to hedge options by first determining whether to buy or sell the underlying asset. When you buy calls or sell puts, you sell the underlying asset. You buy the underlying asset when you sell calls or buy puts. Put options have a negative delta, while call options have a positive.

For example, suppose you buy 15 call option contracts with a delta of 0.2. You are long delta, so you must sell deltas to create a delta-neutral position. Next, you need to find the quantity of the underlying asset you need to hedge. To find the delta hedge quantity, you multiply the absolute value of the delta by the number of option contracts and multiply that by 100 (each option contract controls 100 shares of stock). In this case, the quantity is 300, or equal to (0.20 x 15 x 100). Therefore, you must sell this amount of the underlying asset to be delta neutral.

# ****Questions on particular products:****

**35. What is interest rate risk?**

Interest rate risk is the probability of a decline in the value of an asset resulting from unexpected fluctuations in interest rates. Interest rate risk is mostly associated with fixed-income assets (e.g., bonds) rather than with equity investments. The interest rate is one of the primary drivers of a bond’s price.

**36. What is reinvestment risk?**

Reinvestment risk refers to the possibility that an investor will be unable to reinvest cash flows received from an investment, such as coupon payments or interest, at a rate comparable to their current rate of return. This new rate is called the reinvestment rate.

Zero-coupon bonds (Z-bonds) are the only type of fixed-income security to have no inherent investment risk since they issue no coupon payments throughout their lives.

**KEY TAKEAWAYS**

* Reinvestment risk is the chance that cash flows received from an investment will earn less when put to use in a new investment.
* Callable bonds are especially vulnerable to reinvestment risk because these bonds are typically redeemed when interest rates decline.
* Methods to mitigate reinvestment risk include the use of non-callable bonds, zero-coupon instruments, long-term securities, bond ladders, and actively managed bond funds.

**37. How do interest rate risk and reinvestment risk interact?**

Fixed income securities such as bonds are instruments that typically pay interest, called the coupon, throughout their lifetimes and then return the face value at maturity. There's generally less risk than with other investments that investors will lose all their money when investing in bonds. There are still risks associated with bonds, though, including factors called interest rate risk and reinvestment risk that focus on whether investors could ultimately be worse off due to changing market conditions.

**Tip**

Interest rate risk refers to the danger of a bond losing value because it pays interest rates below what would-be buyers can otherwise find in the market. Reinvestment risk refers to investors not being able to find a similarly paying investment for their proceeds from a bond.

**Exploring Yield vs Price**

A fixed-rate bond has a stated, unchanging coupon payment it disburses every period - for instance, T-Bonds pay interest semi-annually. The total coupon payments for the year divided by the bond price is the annual yield. Coupons don’t change on fixed-rate bonds, but prices do, and thus so do yields.

For an already issued bond to be sellable, its yield must compete with current interest rates. Thus, as rates and yields go up, prices must come down, and vice versa.

**Interest Rate Risk**

The scenario in which interest rates rise after a bond is issued leads to interest rate risk. Since prices will decline if interest rates rise, the holder of a fixed-rate bond may experience a capital loss if the bond is sold before its maturity date. The longer the period until maturity, the more the bond is subject to interest rate risk.

At maturity, the bond will refund the face amount, so bonds near maturity have little interest rate risk. Bond duration is a mathematical equation that signifies how sensitive a bond is to interest rate risk -- bonds with relatively low durations are more resistant to interest rate risk.

**Understanding Reinvestment Risk**

What if interest rates go down instead? The price of a fixed-rate bond will rise and entice some holders to sell the bond for a profit. But others will hold onto the bond and will find that they cannot make as much interest income from reinvesting the periodic coupon payments they receive.

This is reinvestment risk -- if interest rates go down, your interest on interest will decline. This lowers a bond’s yield to maturity, which is a function of the total income, including reinvested interest income, which will be provided by the bond.

**Understanding Call Risk**

Some bonds are considered callable bonds, which means that the organization that issued them can pay them back early through a process known as calling them. Exactly when the bonds can be called depends on their original terms.

Typically, bonds will be called early if the issuer can replace them with lower interest bonds or simply wants to save money on long-term interest payments. This presents a particular type of reinvestment risk, known as call risk, to the investors in the bonds, since the bonds will likely be called when comparable interest rates are no longer available.

**Floating-Rate Bonds**

Some bonds have variable coupons that float with current interest rates. These instruments tend to have stable prices because their coupons remain competitive within the changing interest rate environment. However, if interest rates go down, so will the bond’s coupon, cutting interest income.

This is income risk. In addition, lower interest rates create reinvestment risk, whether the bond is fixed rate or floating rate. Floating rate bonds are suitable for investors who are more sensitive to interest rate risk than to income risk, such as investors who do not plan to hold a bond until maturity.

**38. Which bond has the greatest associated interest rate risk? A five year zero coupon bond? Or a five year bond that pays coupons?**

Answer and Explanation: A five-year zero-coupon bond has more interest rate risk. Interest rate risk is higher for bonds that pay lower coupon payments.

Therefore, bonds with longer maturities generally have higher interest rate risk than similar bonds with shorter maturities. to compensate investors for this interest rate risk, long-term bonds generally offer higher coupon rates than short-term bonds of the same credit quality.

**39. Which is more volatile, a 20-year zero coupon bond or a 20-year 4.5% coupon bond?**

Zero-coupon bonds tend to be more volatile because they do not pay any interest during the life of the bond. These bondholders receive the face value on maturity, thus the only value in these bonds happens closer to maturity.

**40. A stock is selling at $90. A 3-month call with a strike price of $100 is selling for $3.105 with a delta of 0.329. How many call contracts are required to perform a hedge on 1,000 shares of this stock? Would they be bought or sold? What happens if the price of the stock falls to $50?**

I think what you are asking is how many 33 delta call would 1000 shares of the underlying stock delta hedge. If One option is on 100 shares then 33 options or one option on 3300 shares will have a delta of 1000 share. This will give you a "static hedge". That is small moves in the share price will be exactly offset by changes in the value of your short option position. A long stock position will hedge a shot call or a long put position. you would collects 3.105 per share on the 3300 shares or $10,246.5.

If the shares dropped to 50 you would lose $40,000 on the share but would still have the premium of 10,246.5 so your net loss if this all happens at expiration is29,753.5

**41. You have two options with the same underlying strike price. One has an exercise date in three months, one has an exercise date in six months. Which comes with the greatest risk?**

The longer the time to expiry or exercise the greater the risk, and correspondingly the higher the premium.

**42. What’s the maximum potential loss you could incur by selling a put on a stock?**

What is the Maximum Loss Possible When Selling a Put? The maximum loss possible when selling or writing a put is equal to the strike price less the premium received.

**43. What are the risks inherent in an interest rate swap?**

Like most non-government fixed income investments, interest-rate swaps involve two primary risks: interest rate risk and credit risk, which is known in the swaps market as counterparty risk. Because actual interest rate movements do not always match expectations, swaps entail interest-rate risk.

# ****Questions related to regulation:****

**44. How has Basel III changed the treatment of market risk?**

Trading positions often face significant financial loss due to their exposure to volatilities present in underlying market risk factors. As it stands today, the trading book fails to capture the severity of such losses adequately, which has spurred the BCBS to propose a framework for the estimation of the minimum capital requirements for market risk, also known as the Fundamental Review of the Trading Book, more commonly known as FRTB (BCBS, 2013[[1]](https://www.wolterskluwer.com/en/expert-insights/basel-iii-minimum-capital-requirements-for-market-risk-frtb-a-practical-example-for-implementing-the-standardized-approach-part-i" \l "_ftn1), 2016[[2]](https://www.wolterskluwer.com/en/expert-insights/basel-iii-minimum-capital-requirements-for-market-risk-frtb-a-practical-example-for-implementing-the-standardized-approach-part-i" \l "_ftn2), 2017[[3]](https://www.wolterskluwer.com/en/expert-insights/basel-iii-minimum-capital-requirements-for-market-risk-frtb-a-practical-example-for-implementing-the-standardized-approach-part-i" \l "_ftn3)). Moreover, the Basel Committee is currently monitoring and revising the implementation of the market risk standard, and proposing updated methods (BCBS, 2018[[4]](https://www.wolterskluwer.com/en/expert-insights/basel-iii-minimum-capital-requirements-for-market-risk-frtb-a-practical-example-for-implementing-the-standardized-approach-part-i" \l "_ftn4)).

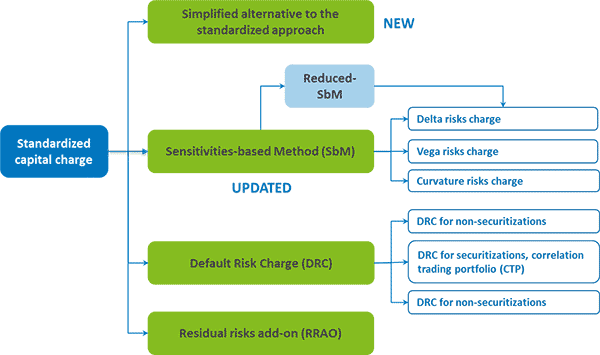
Market liquidity risk plays a key role in both the standardized approach (SA) and the internal model approach (IMA). In the IMA the framework introduces the expected shortfall (ES), substituting the value at risk (VaR) as a measure for the measurement of market risk. The new framework also introduces a profit and loss attribution (PLA) test that the trading desk must pass if they want to implement IMA.

Banks do not only have to estimate capital against the exposure to modellable risk factors. The framework now recognizes an additional capital requirement dedicated to non-modellable risk factors (NMRFs).

To ensure banks do not create regulatory arbitrage, the new framework aims to close the gaps between the treatment of trading and banking book exposures. To this end updated revisions to the boundary between the two books have been proposed by the committee (BCBS, 20184).

**The Standardized Approach to Market Risk**

Banks must devote a series of methods for implementing the standardized approach (SA) (Figure 1): a) the sensitivities-based method (SbM), b) the default risk charge (DRC), and c) the residual risks add-on (RRAO) methods. The committee also proposes a simplified alternative standardized approach to market risk.



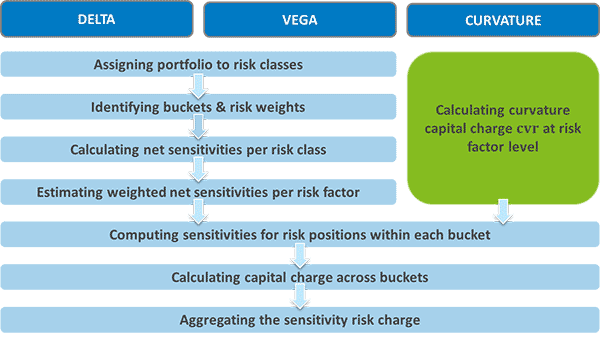
*Figure 1: Methods for implementing the standardized approach*

**Sensitivities-based Method**

The SbM framework suggests that banks use sensitivity analysis for the estimation of capital charges against delta, vega, and curvature risks.

Banks should follow several steps for estimating the capital charges based on SbM. These steps include:

1. The assignment of the portfolio to risk classes;
2. The identification of buckets;
3. The estimation of net sensitivities for each risk class;
4. The calculation of weighted net sensitivities for each risk factor;
5. The computation of sensitivities for risk positions within each bucket,
6. the estimation of capital charge across buckets;
7. The aggregation of the sensitivity risk charge based on correlation scenarios (see Figure 2).



*Figure 2: Main steps for estimating capital charge based on the sensitivities-based method*

Delta and vega risk charges are computed individually for seven risk classes, the capital charges within each bucket are aggregated and finally the capital requirements across those buckets is calculated. The weighted delta and vega sensitivities drive the capital on risk factors (RFs) and the correlation factors with and across buckets.

Banks also need to capture risks assigned to non-linear instruments which means they must estimate curvature risk dealing with the second-order sensitivity measurements. Thus, any changes in the price of an option not identified by delta and vega risk is addressed by curvature risk. The final Basel III framework approximates the curvature as an incremental capital charge above delta capital charge.

After estimating the curvature risk charge, banks have to apply the sensitivity risk charge aggregation based on three scenarios on the correlations between risk factors within a bucket and cross-bucket correlations within a risk class. In fact, the bank has to stress the correlation factors based on three scenarios:

1. A shock of increasing the level of correlations by 25%;
2. No shock, i.e. unchanged correlation;
3. A shock of reducing the level of correlations based on a formula proposed by the framework as proposed in the latest amendments (BCBS, 20184, Annex A: 15 paras 54c).

Banks have to implement the above three scenarios individually for each risk class to calculate the risk charges accordingly. A portfolio exposed to risk classes must aggregate the associated risk charges and the three scenario-based risk charges resulting in three values of the aggregated portfolio. The highest of the three aggregated values is the recognized capital charge at portfolio level (Figure 3).



Figure 3: The three steps for estimating of the sensitivity risk charge at the portfolio level

Grounded by evidential eligibility criteria, a simplified reduced SbA may be an alternative option for banks. The absence of vega estimation and curvature risks, and the reduction of RFs and correlation scenarios under consideration are both benefits of the less demanding framework, however, the upsurge of risk weights under this method means the capital charge rises significantly.

In the latest proposed amendments, further to alternative reduced SbA, the committee recommends a second alternative option whereby a recalibrated version of the Basel II standardized approach can be used (BCBS, 2018). It includes four multiplication scaling factors applied respectively to the capital requirements, estimated by the SA, in the four risk classes: FX risk, commodity risk, general and specific interest rate risk, general and specific equity risk. The over capital requirement results in summing up the recalibrated capital estimations (BCBS, 20184, Annex F: 39 paras 3).

**Example of the Estimation of Delta Capital Charge**

Let us examine a case of a banking institution based in the Euro-zone area that holds a portfolio consisting of three assets:

* A five-year maturity corporate bond with a modified duration of 4.5, denominated in GBP;
* A two-year maturity government bond with a modified duration of 2.8, denominated in GBP; and,
* A seven-year maturity corporate bond with a modified duration of 6.7, denominated in EUR.

By employing SbM the bank estimates delta general interest rate risk (GIRR) capital charges following the steps presented below and as illustrated in Table I:

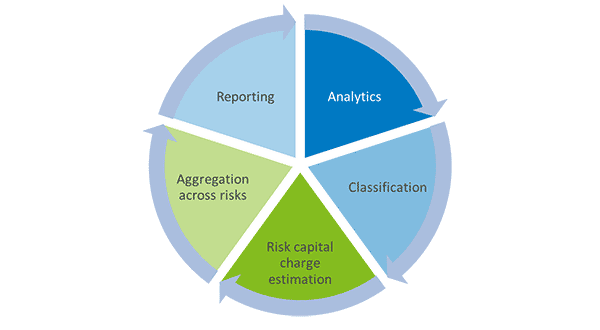


Table I. Steps of estimating the delta (GIRP) capital charge

1. All assets are interest rate sensitive. As a result, they fall under the GIRR risk class;
2. In the view of the bucket definition of GIRR delta the two currencies, that is, GBP and EUR, define two buckets, b1 and b2, accordingly;
3. The sensitivities Sk, Sl and Sm denominated to GBP and EUR are approximated to the modified duration of the instruments, defined at the degrees of 4.5, 2.8 and 6.7 respectively;
4. By knowing the instruments’ maturity, the bank identifies the vertexes, as of 5, 2 and 7, set by the framework;
5. Corresponding to the above vertexes risk weights (RWs), distinct by the framework, are within a range of [0.90% - 1.20%], [1.10%  - 1.50%] and [0.90%  - 1.20%];
6. The weighted sensitivities are estimated (BCBS, 2016[[5]](https://www.wolterskluwer.com/en/expert-insights/basel-iii-minimum-capital-requirements-for-market-risk-frtb-a-practical-example-for-implementing-the-standardized-approach-part-i" \l "_ftn5) 25 paras 67), within a range of [0.041 - 0.054], [0.031 - 0.042] and [0.060 - 0.080], respectively;
7. At the level of each bucket:

- For assets one and two, the correlations between the two weighted sensitivities set to the range of [0.941 - 0.941] as derived from the rules defined by the framework (note that as b2 contains only one asset, a correlation factor is out of consideration);

- For all assets, the risk positions are calculated resulting of a range within [0.070 - 0.095] for b1, and [0.004 - 0.006] for b2;

8. The sensitivities for all risk factors for b1 are estimated within a range of [0.0713 - 0.096] for assets one and two, whereas asset three belongs to bucket b2 calculated within a range of [0.0603 - 0.0804].

9. The level of correlation across those buckets is defined by the framework and set to a degree of 0.5;

10. The delta capital charge estimated to the range of values between 0.096 and 0.129.

**Layout of a process for implementing Basel III minimum capital requirements for market risk**

In conclusion, initially banks must apply the necessary analytics for estimating the market risk sensitivities, classify the risk exposures and the assets under study to identify the associated risk weights, calculate the risk capital charge based on the formulas provided by the framework, apply aggregation rules within and across buckets, report associated capital against risk and losses. The cycle process of implementing Basel III minimum capital requirements for market risk based on the standardized approach is illustrated in Figure 4.

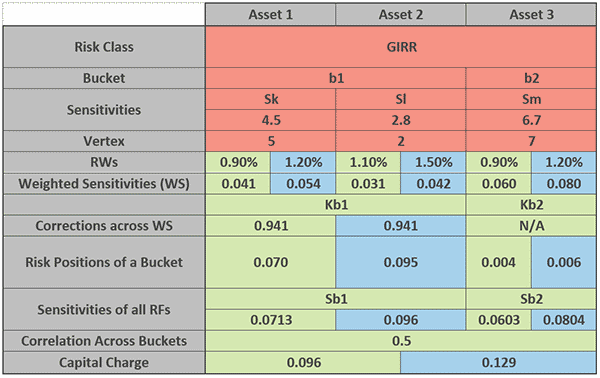


Figure 4: Process steps of implementing Basel III minimum capital requirements for Market Risk

**45. What the implications of Basel IIIs new trading book rules for market risk professionals?**

In the wake of the Lehman Brothers collapse of 2008 and the ensuing financial crisis, the BCBS decided to update and strengthen the Accords. The BCBS considered poor governance and risk management, inappropriate incentive structures, and an overleveraged banking industry as reasons for the collapse. In November 2010, an agreement was reached regarding the overall design of the capital and liquidity reform package. This agreement is now known as Basel III.

Basel III is a continuation of the three pillars along with additional requirements and safeguards. For example, Basel III requires banks to have a minimum amount of common equity and a minimum liquidity ratio. Basel III also includes additional requirements for what the Accord calls "systemically important banks," or those financial institutions that are considered "too big to fail." In doing so, it got rid of tier 3 capital considerations.

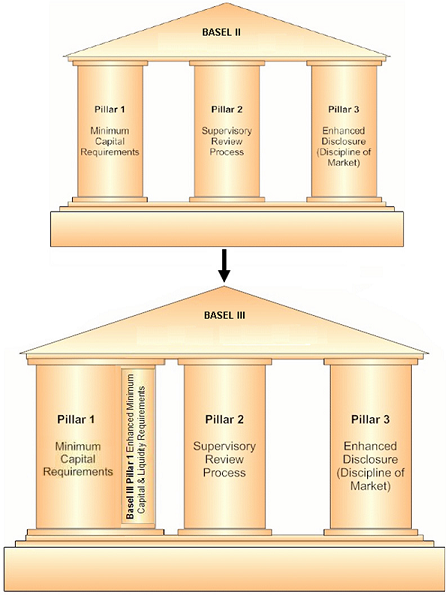
The Basel III reforms have now been integrated into the consolidated Basel Framework, which comprises all of the current and forthcoming standards of the Basel Committee on Banking Supervision. Basel III tier 1 has now been implemented and all but one of the 27 Committee member countries participated in the Basel III monitoring exercise held in June 2021. The final Basel III framework includes phase-in provisions for the output floor, which will start at 50% on Jan. 1, 2023, rising in annual steps of 5% and be fully phased-in at the 72.5% level from January 2028. These 2023 onward measures have been referred to as Basel 3.1 or Basel IV.

Basel III is an extension of the existing Basel II Framework, and introduces new capital and liquidity standards to strengthen the regulation, supervision, and risk management of the whole of the banking and finance sector.

It was agreed upon by the members of the Basel Committee on Banking Supervision in 2010–2011, and was scheduled to be introduced from 2013 until 2015. However, changes made from April 2013 extended implementation until March 31, 2018. The Basel III requirements were in response to the deficiencies in financial regulation that is revealed by the 2000’s financial crisis. Basel III was intended to strengthen bank capital requirements by increasing bank liquidity and decreasing bank leverage.

The global capital framework and new capital buffers require financial institutions to hold more capital and higher quality of capital than under current Basel II rules. The new leverage ratio introduces a nonrisk-based measure to supplement the risk-based minimum capital requirements. The new liquidity ratios ensure that adequate funding is maintained in case there are other severe banking crises.

The figure below shows how Basel III strengthens the three Basel II pillars, especially Pillar 1 with enhanced minimum capital and liquidity requirements. Figure 1. Basel II and Basel III pillars



**Capital requirements**

The Basel III rule introduced the following measures to strengthen the capital requirement and introduced more capital buffers:

**Capital Conservation Buffer** is designed to absorb losses during periods of financial and economic stress. Financial institutions will be required to hold a capital conservation buffer of 2.5% to withstand future periods of stress, bringing the total common equity requirement to 7% (4.5% common equity requirement and the 2.5% capital conservation buffer). The capital conservation buffer must be met exclusively with common equity. Financial institutions that do not maintain the capital conservation buffer faces restrictions on payouts of dividends, share buybacks, and bonuses.

**Countercyclical Capital Buffer** is a countercyclical buffer within a range of 0% and 2.5% of common equity or other fully loss absorbing capital is implemented according to national circumstances. This buffer serves as an extension to the capital conservation buffer.

**Higher Common Equity Tier 1 (CET1)** constitutes an increase from 2% to 4.5%. The ratio is set at:

* 3.5% from 1 January 2013
* 4% from 1 January 2014
* 4.5% from 1 January 2015

**Minimum Total Capital Ratio** remains at 8%. The addition of the capital conservation buffer increases the total amount of capital a financial institution must hold to 10.5% of risk-weighted assets, of which 8.5% must be tier 1 capital. Tier 2 capital instruments are harmonized and tier 3 capital is abolished.

**Leverage ratio**

Basel III introduced a minimum "leverage ratio". The leverage ratio was calculated by dividing Tier 1 capital by the bank's average total consolidated assets; the banks were expected to maintain a leverage ratio in excess of 3% under Basel III. In July 2013, the US Federal Reserve Bank announced that the minimum Basel III leverage ratio would be 6% for 8 SIFI banks and 5% for their bank holding companies.

**Liquidity requirements**

Basel III introduced two required liquidity ratios:

* **Liquidity Coverage Ratio (LCR)** ensures that sufficient levels of high-quality liquid assets are available for one-month survival in a severe stress scenario.
* **Net Stable Funding Ratio (NSFR)** promotes resilience over long-term time horizons by creating more incentives for financial institutions to fund their activities with more stable sources of funding on an ongoing structural basis.

**Changes to Counterparty Credit Risk (CCR)**

Basel III introduced capital requirements to cover Credit Value Adjustment (CVA) risk and higher capital requirements for securitization products.

**II**

The Fundamental Review of the Trading Book (“FRTB”) was initiated by the Basel Committee on Banking Supervision (“BCBS”) in the years following the Great Financial Crisis (“GFC”) of 2007-2009, with the aim of completely revising the approach to calculating risk-based capital requirements for trading activities (i.e., “market risk capital”).

As we will discuss in forthcoming blogs, this complex series of reforms will likely have far-reaching impacts not only on the trading business models of large banks, but also on liquidity provision in key funding markets and therefore the ability of certain non-financial end-users to raise funds.  These impacts are inevitable given that 73 percent of the funding for U.S. non-financial corporations is generated by the U.S. capital markets[[1]](https://www.sifma.org/resources/news/the-fundamental-review-of-the-trading-book-frtb-an-introductory-guide/#_ftn1), and because banking organizations remain crucial providers of capital markets services to non-financial end-users.  Given these potentially far-reaching impacts, it is important for us to first understand what the FRTB was designed to do, why we are hearing more about it now, and what is contained in the package of reforms.

**What were regulators’ goals in undertaking the FRTB?**

In the immediate wake of the GFC, the BCBS introduced the so-called “Basel II.5” package of reforms, which included a significant increase in the market-risk capital standards for banks. While increasing the overall quantum of capital required for market risk and making other modifications to the calculation of market risk capital (particularly the introduction of the Incremental Risk Charge and the Stressed Value-at-Risk “VaR”) , the reforms were seen mostly as a stop gap measure until the underlying structural problems with market risk capital standards could be properly addressed by the Committee, which would come later in the form of the more comprehensive FRTB.

Those structural problems included the lack of a clearly defined boundary between the trading book and the banking book, which provided opportunities for arbitrage between books to obtain more favorable capital treatment for specific instruments or portfolios. Another important concern involved the weaknesses associated with the existing VaR approach to modelling risk. While this approach did a good job of modeling risk under normal market conditions, it performed poorly during periods of market volatility, such as those that occurred during the GFC (i.e., put differently, it did a poor job of capturing so-called “tail risks”). The addition of a “stressed VaR” measure in the Basel II.5 framework was seen as insufficient to compensate for these weaknesses.

A further structural deficiency of the existing market risk capital framework was its failure to consider the relative liquidity of trading book positions and the risks of market illiquidity.  The existing framework was based on liquidity horizon assumption of ten days, ignoring the fact that many positions may take far longer to liquidate in practice (and particularly during periods of market stress).  Other identified problems with the existing framework included a lack of transparency and comparability between the internal models and standardized approaches, particularly in areas such as hedging and diversification, as well as the more favorable treatment of credit risk in the trading book, which had led banks putting an excessive number of their credit positions in the trading rather than the banking book.

To address these deficiencies, the BCBS issued a consultation on the FRTB in 2012. This was followed by further consultation exercises in 2013 and 2014, an initial set of standards in 2016 and subsequent modifications to those standards that were ultimately finalized in early 2019 (see timeline in Figure 1 below).

Throughout, the core goals of the FRTB initiative were to:

a) create a clear regulatory boundary between the trading and banking books;

b) replace the VaR approach to risk measurement with a more comprehensive alternative known as Expected Shortfall (“ES”);

c) revise the Standardized Approach (“SA”) to make it more risk-sensitive and allowing it to act as a credible fallback Internal Models Approach (“IMA”);

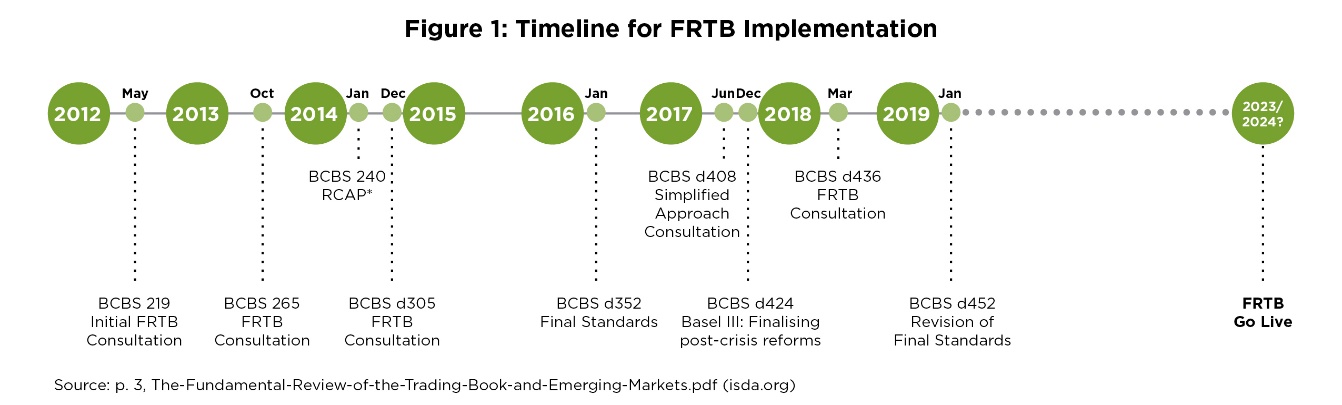
d) replace the static 10-day liquidity horizon assumed under the VaR framework with varying liquidity horizons in the IMA;

e) introduce a new capital add-on for risk factors that fail modelability tests, known as Non-Modellable Risk Factors (“NMRFs”); and

f) create a new and more robust approvals processes for obtaining regulatory approval for IMA use and requiring these approvals to occur at the level of individual desks rather than granting firm-wide approval.

Although not a stated objective of the BCBS, industry quantitative impact study (“QIS”) exercises have demonstrated[[2]](https://www.sifma.org/resources/news/the-fundamental-review-of-the-trading-book-frtb-an-introductory-guide/#_ftn2), that the market risk changes will also likely increase the aggregate level of market related RWA in the US banking system if implemented without any modifications.  This impact would be distributed unevenly, imposing a significant capital penalty on banks with large trading books. Depending on how it is implemented, the FRTB could also have a disproportionate impact on some types of products/markets versus others. For example, the way securitization exposures are calculated under the Basel FRTB standards could discourage market making in that sector. Similarly, less liquid markets (e.g., certain corporate bonds and emerging markets) could also see a reduction in market making activity; this is because these types of products/markets are more like to fail the FRTB’s modelability tests, resulting in more punitive capital treatment under the Basel standards. We will discuss these potential impacts among others in future blogs.

**When will the FRTB be implemented?**



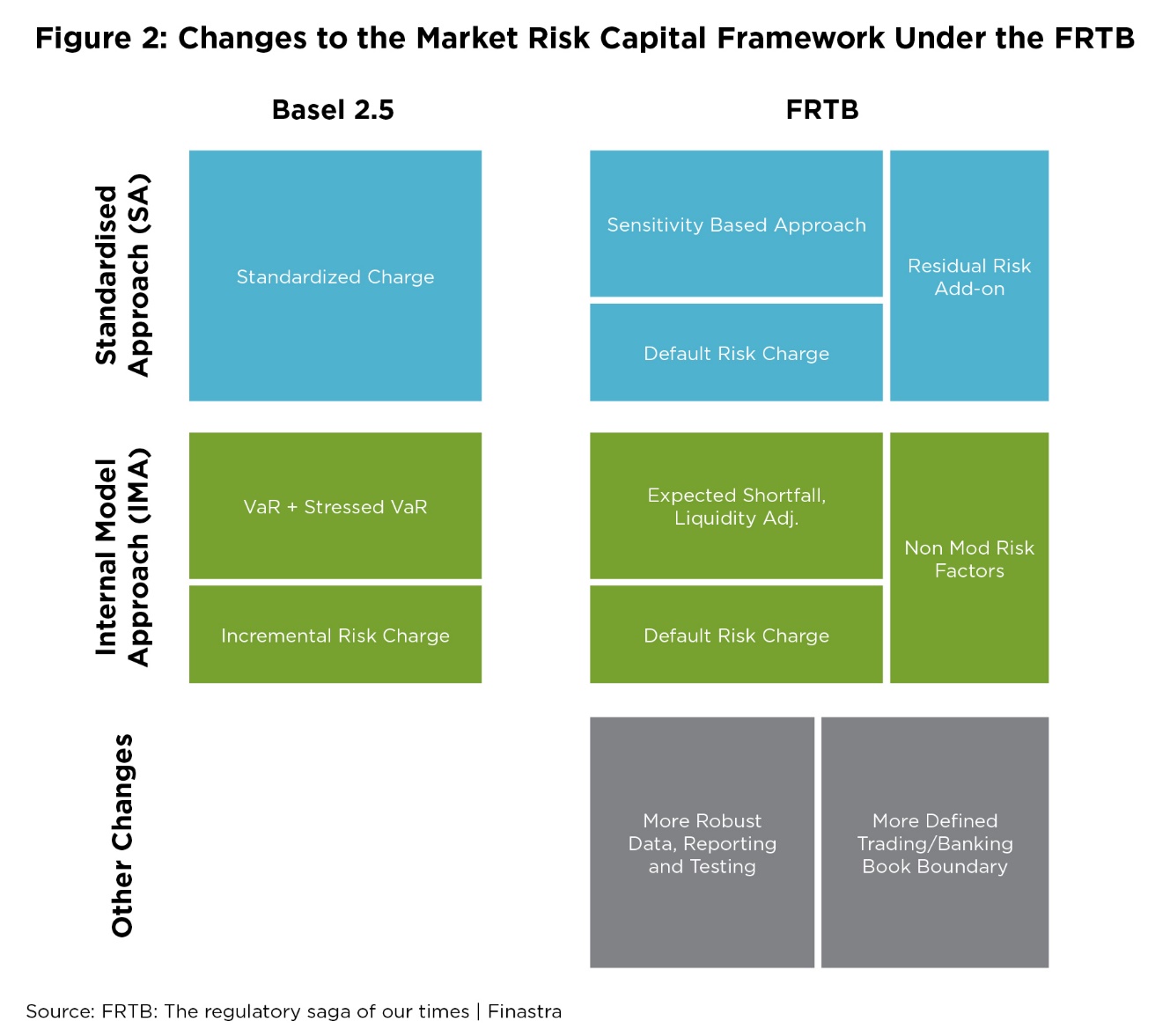
While the global standards have been finalized at the Basel level, important outstanding questions remain about the implementation of the final rules in each jurisdiction. Perhaps the most obvious uncertainty concerns the timing of FRTB implementation.  The final BCBS standards set January 1, 2022, as the deadline for national implementation, but in March of last year the BCBS agreed to push back that date to January 1, 2023, to give banks additional time to respond to the COVID-19 crisis.  Given the complexity of the FRTB changes (along with the remaining Basel III Endgame reforms to other RWA requirements), it was widely assumed that proposed rulemakings would be issued in all the major jurisdictions by the end of 2020 at the latest.  This timeframe was seen as necessary to provide for significant public input on the proposed changes and to ensure that there was an adequate amount of time between finalization of national rules and the final go-live date for banks to build, test and validate the necessary systems and models required to implement the FRTB.

However, as of late August 2021, only a few jurisdictions (such as Hong Kong) appear to be on track for full implementation by the 2023 deadline.  The EU has issued implementing regulations relating to the FRTB, with standardized reporting requirements due to come into effect by September 2021 and full implementation pushed out to 2024.  Of note, the U.S. banking regulators had yet to issue a formal FRTB proposal, or indeed a proposal to implement other elements of the Basel III Endgame reforms, having focused much of their attention over the past year on dealing with the response to the pandemic.  This strongly suggests that the timeline for U.S. implementation (and likely most other jurisdictions) will extend into 2024 or possibly later.  While a further delay is viewed as almost inevitable, neither the BCBS nor U.S. regulators have, as of this writing, confirmed a new implementation date for the FRTB.

Although much of the focus has been on the implementation timeline, important substantive issues remain undecided.  While the FRTB is an internationally agreed minimum standard, national regulators are afforded flexibility and discretion in how they implement it (discretion that has often been used by U.S. regulators to “gold plate” other Basel Accords).  In recent public comments made at the BPI-SIFMA Prudential Conference[[3]](https://www.sifma.org/resources/news/the-fundamental-review-of-the-trading-book-frtb-an-introductory-guide/#_ftn3), Federal Reserve General Counsel Mark Van Der Weide committed to a “robust and faithful” implementation of the FRTB and the related Basel III endgame proposals, but also acknowledged that the agencies retained the flexibility to modify the standards to fit unique U.S. legal requirements and business/market structures, as well as to determine the scope of applicability of the reforms.  We will discuss some of the outstanding substantive issues later in this blog series.

**What are the core components of the FRTB?**

The finalized BCBS framework outlines two approaches that firms can adopt to calculate their market risk capital requirements – the Standardized Approach (“SA”) and Internal Models Approach (“IMA”).  All banks must implement the SA; however, some firms may additionally opt for the IMA subject to initial and continued conformance to stringent model performance standards.  Consistent with the goal of establishing a clearer boundary between the credit and trading books, the final rule also significantly revised the approach to classifying firm positions as either trading or banking exposures.  Below is a high-level description and graphic of each of the key elements of the final FRTB standards.



***Data, Reporting and Testing Changes***

In the FRTB, the BCBS makes changes to the way market risk data is gathered, reported, monitored, and verified for accuracy. At a high-level, these changes include:

* Implementing a more granular desk level model review and approval process;
* Requiring desk level profit and loss attribution tests (“PLAT”) and back testing to be performed on a daily basis;
* Expanding the substance of intra-day monitoring and measurement of market risk; and
* Applying more granular assessments of model performance.

***Trading Book Definitions***

As noted above, one of the main concerns about the existing market risk framework was that it was insufficiently prescriptive in its classification of trading instruments, allowing for significant arbitrage to occur between the trading and banking books, and leading to inconsistent interpretations between firms.  The BCBS addressed their concerns in the FRTB framework by:

Defining a new boundary based on whether the bank *intends*to trade an asset or hold it to maturity;

Providing a presumptive list of trading book assets and Regulatory Trading Desks (“RTDs”);

Imposing more restrictive rules for internal transfers between bank and trading booking, including prohibiting any capital relief as a result of reclassification;

Placing a restriction on recognition of internal risk transfers (only external risk transfers are recognized under the framework); and

Imposing a more robust framework for hedging recognition to focus on their effectiveness during periods of stress.

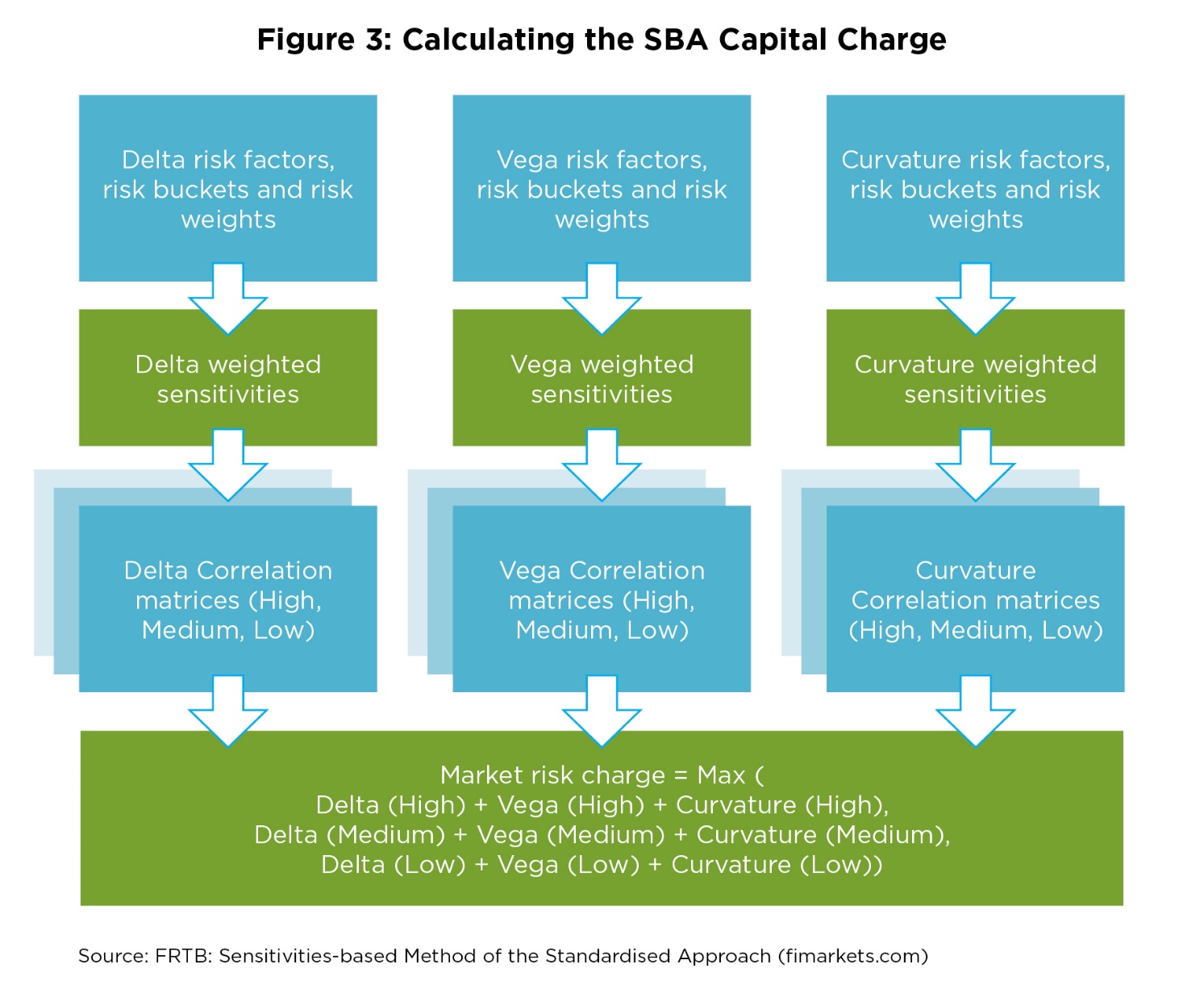
***Standardized Approach (SA)***

The SA is the default market capital requirement for all banks that fall within the scope of the FRTB, regardless of whether a firm pursues IMA accreditation for the permitted asset classes on a desk-by-desk basis.  Importantly, because the IMA approach is subject to initial and continual performance testing, the SA serves as a fallback if a firm fails to meet the performance goals and is the only approach available for correlation trading.[[4]](https://www.sifma.org/resources/news/the-fundamental-review-of-the-trading-book-frtb-an-introductory-guide/#_ftn4) As noted above, the SA is significantly more risk sensitive than current standardized approaches.  However, that enhanced risk sensitivity – which occurs primarily through the introduction of a new Sensitivities-Based Approach (“SBA”) – is reliant on data obtained from a bank’s pricing models to derive capital requirements. That means that many firms will have to build significant new modelling capacities to meet the requirements.

The SA capital risk charge is comprised of three components: the SBA, the Default Risk Charge (“DRC”) and the Residual Risk Add-on (“RRA”).  The SA capital charge is calculated using the sum of the SBA, DRC and, if applicable, the RRA charge.

**Sensitivities-Based Method (“SBA”)**

The SBA is a form of Parametric VaR with regulatory weights and correlations.  Banks are required to compute risk factors (i.e., observable or measurable market data that is likely to influence the valuation and therefore the profit and loss (“P&L”) generated by a financial instrument) sensitivities (Delta, Vega, and Curvature)[[5]](https://www.sifma.org/resources/news/the-fundamental-review-of-the-trading-book-frtb-an-introductory-guide/#_ftn5) for seven regulatory defined risk classes.[[6]](https://www.sifma.org/resources/news/the-fundamental-review-of-the-trading-book-frtb-an-introductory-guide/" \l "_ftn6)  These risks are then aggregated across buckets, which are sets of instruments of the same risk class sharing the same characteristics and therefore a similar risk profile (a bucket may mean, in this context, a currency or commodities).  Moreover, to capture changes in correlations due to financial stresses, banks are required to use three different correlation scenarios (low, medium, and high) to calculate three separate risk charges for each of the regulatory defined risk classes.  Following the calculation methodology, the firm sums the all the risk class charges and then uses the largest as the SBA charge (see Figure 3 below for a visual representation of this process).



**Default Risk Charge (“DRC”)**

The framework includes a DRC to capture “jump-to-default risk” i.e., credit spread risk.  Only certain portfolio types are exposed to this type of risk, so instead of seven risk classes under the SBA, the risk classes are reduced to three under the DRC: debt instruments, equity products, and securitizations.  To determine the aggregate DRC, firms are required to apply differing risk weights, formulas, and expectations regarding netting across the three asset classes.

**Residual Risk Add-on (“RRA”)**

The final component of the SA is the RRA, which is designed to capitalize exposures that are not adequately captured by the SBA.  Generally, this charge is applied to more complex transactions where the exposure is difficult to quantify (e.g., products in incomplete markets, products having correlation risk).  The methodology applies a simplistic formula that aggregates all residual instruments’ gross notional amount multiplied by a risk weight of 0.1 percent of or in the case of exotic (i.e., highly complex) instruments, 1.0 percent.

***The Internal Models Approach (IMA)***

The IMA fundamentally changes the approach to modeling market risk by reforming the measurement methodology and expanding model performance requirements.  Most importantly, IMA replaces the long standing VaR-based approach for quantifying market risk with an Expected Shortfall (“ES”) approach.  As noted above, while VaR does a good job of capturing risk in normal markets, an ES approach is more effective in capturing the “fatter tail” distributions of risk typical of stressed markets when multiple asset classes move in tandem.

The IMA also removes the Incremental Risk Charge introduced in Basel 2.5 and replaces it with a Default Risk Charge (referred to here as an “IMA DRC”), which acts (as it does under the SA) to capture losses that stem from an obliger defaulting.  It places limits on the benefits of hedging, netting and diversification and no longer permits correlation trading positions capital to be estimated with internal models.  Lastly, the framework implements asset class-specific liquidity horizons which more rigorously incorporates the impact of stress periods; and introduces the concept and approach for non-modellable risk factors.

The use of internal models and their on-going performance is subject to considerable supervisory scrutiny under the FRTB as well as subject to quantitative requirements.  Internal models are also now subject to regulatory approval at the trading desk level. For a desk to qualify for IMA, they must demonstrate that a model is supported by adequate P&L attribution tests and back testing results.

**P&L attribution** is not a new concept to market risk management; however, it is new to the capital framework in the level of specificity that is included in the new framework.  For initial and continued approval, the framework requires daily comparison and analysis of P&L statements generated by a bank’s front-office, also known as hypothetical P&L (“HPL”), with the results produced by middle/back office, also known as risk-theoretical P&L.  The framework applies quantitative limitations on the amount of unexplained P&L and the variability of the unexplained between the two systems.  Failure to meet those requirements results in loss of IMA approval and application of the standardized approach.

Bank testing requirements are equally stringent.  The framework requires a comparison of the VaR measure calibrated to a one-day holding period against each of the Actual P&L (“APL”) and HPL over the prior 12 months.  Specific requirements to be applied at the bank-wide level and trading desk level are set out by the BCBS and back testing of the bank-wide risk model must be based on a VaR measure calibrated at a 99th percentile confidence level.  Finally, the scope of the portfolio subject to bank-wide back testing must be updated quarterly based on the results of the latest trading desk-level back testing, risk factor eligibility and P&L attribution tests.  Again, failure to meet the performance requirements will result in the loss of IMA status.

In short, these new requirements make qualifying for use of IMA an extraordinarily difficult and resource intensive exercise.  For those trading desks that do succeed in obtaining approval to use this approach, there are three components that are used to calculate the IMA capital charge: Expected Shortfall or ES; modellable risks and non-modellable risks; and a default risk charge.  For desks where the bank cannot or chooses not to pursue IMA qualification, they must calculate capital charges using the SA.

**Expected Shortfall (“ES”) Method**

The ES and VaR are both used to measure portfolio risk.  However, the ES is a significantly more conservative measure, since it measures the expected value of all changes in the portfolio value in the tail of the P&L distribution that exceed the VaR.  To account for this additional conservatism, the BCBS lowered the confidence interval from 99 percent requirement under the VaR approach to 97.5 percent under the ES.  Additionally, the ES must be calculated for a maximum stress period over the bank’s *total history of observations* verses the current look back period of one year or 270 days.  Consequently, this approach, while superior to VaR for capturing tail risk, is also far more data and resource intensive.

**Modellable and Non-Modellable Risk Factors (“NMRF”)**

The path to determining the capital charge under the IMA is dependent on the ability of a firm to model risk factors.  A risk factor’s model-ability is determined by a series of quantitative criteria regarding the frequency, observability, and durability of a risk factor’s pricing information.  Importantly, the firm must continually assess the eligibility of these risk factors to be included in the IMA models.  If a firm cannot or no longer can support the model-ability of a risk factor, the firm must apply a capital add on which is calibrated under a stress scenario. Instances of NMRF are more likely in more thinly traded and less liquid markets, and the issue of how NMRF are defined continues to be a subject of significant debate (and will be discussed in a future blog).

**IMA Default Risk Charge (“IMA DRC”)**

While migration risk is captured in ES, the IMA framework separately capitalizes default risk. The IMA DRC replaces the Incremental Risk Charge and will apply to credit positions as well as equity positions.  The IMA DRC will also include a floor for the probability of default.

**Conclusion**

The FRTB represents a sweeping overhaul of the way banks calculate their trading risk capital charges and will have wide ranging impacts on the business models of banking organizations and funding markets for many years to come. As U.S. regulators move closer to a proposed rulemaking, we will release a series of blogs on some important outstanding issues that still need to be addressed prior to implementation. These include ongoing concerns around the design of the P&L attribution tests, the NMRF framework, and the interaction between the FRTB and other existing domestic capital requirements.

**46. How could the Basel III treatment of trading books be improved?**

**Shortcomings of previous regime:**

**1. Shortcomings of the framework exposed by the financial crisis**

The recent crisis exposed material weaknesses in the capital treatment of banks’ trading activities. Some of the most pressing deficiencies of the trading book regime were addressed by the July 2009 revisions to the market risk framework, while others have been dealt with as part of Basel III. However, the Committee has agreed that a number of the market risk framework’s fundamental shortcomings remain unaddressed and require further attention. The Committee has agreed that the future trading book regime must address the weaknesses set out below, which are discussed in more detail in Annex 1. The crisis and pre-crisis experience highlighted a number of shortcomings in the trading book regime. These can be broadly categorised into weaknesses arising from:

(a) The overall design of the regulatory capital framework, especially the inclusion of instruments exposed to credit risk in the trading book;

(b) The risk measurement methodologies used under the models-based and standardised approaches; and (c) The valuation framework applied to traded instruments.

In combination, these shortcomings resulted in materially undercapitalised trading book exposures prior to the crisis.

**1.1 Weaknesses in the design of the regulatory capital framework**

While the undercapitalisation of trading book exposures has often been the result of the methodologies used for risk measurement and valuation (both of which are discussed later in this section), elements of the overall design of the regime also contributed to, and amplified, the problems exposed during the crisis. These include:

• **The role of the regulatory boundary:** The Committee believes that its definition of the regulatory boundary has been a key source of weakness in the design of the current regime. A key determinant of the boundary is banks’ intent to trade, an inherently subjective criterion that has proved difficult to police and insufficiently restrictive from a prudential perspective in some jurisdictions. Coupled with large differences in capital requirements against similar types of risks across either side of the boundary, the capital framework proved susceptible to arbitrage. For example, prior to the crisis, it was advantageous for banks to classify an increasing number of instruments as “held with trading intent” (even if there was no evidence of regular trading of these instruments) in order to benefit from lower trading book capital requirements. During the crisis the opposite movement of positions from the trading book to the banking book was evident at times in some jurisdictions.

• **The lack of credible options for the withdrawal of model approvals**: The design of the current framework does not embed a clear link between the models-based and standardised approaches either in terms of calibration or in terms of theconceptual approach to risk measurement. In part as a consequence of this, a key weakness of the design of the current framework has been the lack of credible options for the withdrawal of model approval. This can be a particular problem in stress periods, where supervisors witness a deterioration in model performance at the same time as raising new capital becomes very difficult.

**1.2 Weaknesses in risk measurement**

In addition to the flaws in the overall design of the framework, risk measurement under both the models-based and the standardised approaches proved wanting:

• **Shortcomings of the models-based approach**: The metric used to capitalise trading book exposures was the 10-day value-at-risk (VaR) computed at the 99th percentile, one-tailed confidence interval. By construction, this is a measure aimed at capturing the risk of short-term fluctuations in market prices. While a 10-day VaR might be useful for day-to-day internal risk management purposes, it is questionable whether it meets the objectives of prudential regulation which seeks to ensure that banks have sufficient capital to survive low probability, or “tail”, events. Weaknesses identified with the 10-day VaR metric include:

1. its inability to adequately capture credit risk;
2. its inability to capture market liquidity risk;
3. the provision of incentives for banks to take on tail risk;
4. and, in some circumstances, the inadequate capture of basis risk.

Perhaps more fundamentally, the models-based capital framework for market risk relied on a bank-specific perspective of risk, which might not be adequate from the perspective of the banking system as a whole. The pro-cyclicality of VaR-based capital charges based on recent historic data and the large number and size of backtesting exceptions observed during the crisis serve to highlight regulatory concerns with continued reliance on VaR.

• **Shortcomings of the standardised approach**: Although the crisis largely brought to the fore problems with the models-based approach to market risk, the Committee has also identified important shortcomings with the standardised approach. These include a lack of risk sensitivity, a very limited recognition of hedging and diversification benefits and an inability to sufficiently capture risks associated with more complex instruments.

**Weaknesses in valuation practices**

The recent crisis highlighted the importance of robust valuation practices, especially of complex or illiquid financial instruments, in times of stress. Different valuation methodologies can have a very material impact on estimated capital resources. Therefore, in assessing capital adequacy, supervisors need to be confident that valuation methodologies are in line with prudential objectives. It is at least as important to have prudent, reliable and comparable estimates of capital resources as to have prudent, reliable and comparable estimates of capital requirements. The crisis highlighted key weaknesses in the valuation framework, including the lack of application of prudent valuation adjustments and the emergence of valuation uncertainty as a key source of solvency concerns.

The 2009 revisions to the market risk framework (“Basel 2.5”) The key elements of these revised market risk standards were:

• **The introduction of the IRC:** In recognition of the fact that the 10-day VaR metric does not sufficiently capture banks’ exposures to credit risk, the 2009 amendments introduced an additional capital charge intended to capture both default risk and credit rating migration risk. The IRC is estimated based on a one-year capital horizon at a 99.9 percent confidence level, consistent with the treatment of credit exposures in the banking book. However, it also takes into account the liquidity of individual instruments or sets of instruments. Unlike the banking book treatment of credit risk, it allows banks to estimate their own asset value correlation parameters.

• **The introduction of stressed VaR**: In addition to the 10-day VaR requirements, the 2009 amendments require banks to calculate a “stressed VaR” measure. The stressed VaR is intended to replicate a VaR calculation that would be generated on the bank’s current portfolio if the relevant market factors were experiencing a period of stress. It should be based on the 10-day, 99th percentile, one-tailed confidence interval VaR measure, with model inputs calibrated to historical data from a continuous 12-month period of significant financial stress. The introduction of stressed VaR is intended, in part, to dampen the cyclicality of the VaR measure and to mitigate the problem of market stresses falling out of the data period used to calibrate the VaR after some time.

• **Alignment of the treatment of securitisation exposures across the banking book and the trading book**: As of July 2009, the Committee as a whole had not agreed that modelling methodologies used by banks adequately captured the risks of securitised products. As a result, it agreed to apply the standardised capital charges based on the banking book risk weights to these exposures. However, the Committee agreed on a limited exception for certain correlation trading activities, where banks are allowed by their supervisor to calculate capital charges based on the CRM. This new model is subject to a strict set of minimum requirements, including the regular application of specific, predetermined stress scenarios and a floor expressed as a percentage of the charge applicable under the standardised approach.

• **Improved risk factor coverage of internal models**: Banks are now explicitly required to incorporate all risk factors in their VaR models that are deemed relevant for pricing purposes, or to justify their omission. Basis risks are also expected to be captured by banks to the satisfaction of the supervisor, as well as event risk (not covered in IRC), which must be included in the VaR measurement. Banks can no longer rely on a surcharge model to capture these risks.

• **Enhanced prudent valuation guidance**: The Committee extended the scope of the prudent valuation guidance to all instruments subject to fair value accounting, including those in the banking book. The Committee also clarified that regulators retain the ability to require adjustments to the current value beyond those required by financial reporting standards, in particular where there is uncertainty around the current realisable value of an instrument due to illiquidity. This guidance focuses on the current valuation of the instrument and is a separate concern from the risk that market conditions and variables might change before the instrument is liquidated (or closed out).

In December 2010, the Committee issued the Basel III rules text, covering details of reforms to bank regulatory standards agreed by the Governors and Heads of Supervision and endorsed by the G20 Leaders earlier that year. Three changes of the Basel III package relate to the capital treatment of trading activities and market risk:

• **Capital charges against credit valuation adjustment (CVA) volatility risk**: The Committee made a number of amendments to strengthen the counterparty credit risk framework. Among the most important elements of the reform package was a requirement that banks be subject to a capital charge against potential mark-to-market losses associated with deterioration in the creditworthiness of a counterparty (CVA risk). Most of the affected instruments, such as OTC derivatives and securities financing transactions (SFTs), are held in the trading book.

• **Treatment of unrealised gains and losses**: Under the changes to the definition of capital, unrealised gains and losses will no longer be filtered out of Common Equity Tier 1 capital. This means that changes to the valuation of all financial instruments held at fair value for accounting purposes will flow directly through to regulatory capital resources.

• **Eligible capital for trading book risks**: As part of the general improvements in the quality of eligible regulatory capital, Tier 3 capital, previously available to meet market risks, will no longer form part of the regulatory capital structure.

**Drawbacks of the current market risk regime**

The July 2009 amendments to the market risk framework were judged by the Committee to be an essential immediate response to the severe undercapitalisation of banks’ trading books. But from the onset, the Committee also recognised the need for initiating a longer term, fundamental review of the risk-based capital framework for trading activities. In part this is because the current treatment of market risk exposures, while a material improvement relative to the previous regime, does not address all of the shortcomings highlighted in Annex 1 and suffers from a number of drawbacks:

• **The framework lacks coherence**: The current framework does not have a single, overarching view of how trading risks should be categorised and capitalised, leading to the concern that some capital charges appear overlapping, for example, the additive approach taken for VaR and stressed VaR. Moreover, the diverse array of capital charges within the amended framework requires the development and validation of several distinct sets of models. These not only require a substantial amount of bank resources to maintain but have also put a severe strain on supervisory oversight.

• **The boundary issue has not been fully addressed**: The July 2009 revisions to the market risk framework made only minor amendments regarding the set of products that should be excluded from the trading book. However, securitisation exposures other than those eligible for the correlation trading portfolio are treated broadly consistently across the regulatory boundary in the 2009 revisions. In spite of those amendments, similar risks continue to be treated differently across the balance sheet. For example, interest rate risk is only capitalised under the Pillar 1 regime if the bank runs this risk in its trading book. Differences in capital requirements across the regulatory boundary can foster incentives for banks to shift instruments to the regulatory regime that treats them more favourably. Where the boundary is not well monitored, banks could act upon those incentives.

• **Market liquidity risk is not evenly captured**: Although the July 2009 revisions introduce elements that better capture market liquidity risk, they are not comprehensive or complete. The IRC and CRM metrics introduce the concept of varying liquidity horizons to account for the fact that banks might be unable to exit risk positions in short time periods due to market illiquidity. But the IRC and CRM cover mainly credit-related exposures and focus on default and credit rating migration risk. Similarly, stressed VaR implicitly captures variations in liquidity premia in times of stress. However, stressed VaR is still based on a 10-day holding period which is, almost by definition, insufficient to capture the risks associated with market illiquidity. Moreover, stressed VaR implicitly assumes that the markets most likely to turn illiquid in the future are those that turned illiquid in a previously observed period of stress.

• **The bank-specific notion of risk is upheld**: Many of the new approaches are still based on a bank-specific view of risk. For example, stressed VaR still relies on an implicit assumption that all banks can exit or hedge their risks within a 10-day horizon, which was not the case in the recent crisis as many banks tried to exit risk positions simultaneously.

**• Standardised approach problems remain unaddressed**: The July 2009 revisions to the market risk framework did not fundamentally change the standardised approach for market risk. The revisions did adjust some risk weights for equity specific risk and required banking book risk weights for the capitalisation of specific interest rate risk in securitisations. But the structural shortcomings of the standardised approach remain unaddressed.

• **There remains a lack of credible options for withdrawal of model approval**: Aside from multipliers on VaR and stressed VaR, there are limited options for supervisors to deal with poorly-specified internal models. The approaches adopted to backstop the CRM (standardised floor and supplemental capital add-ons from prescribed stress tests) suggest possible alternatives for limiting the reliance on models. The evaluation of backtesting results also suggests a need for regulators to determine specific areas of imprecision, versus focusing on the top-of-the-house risk measure.

• **The relationship between the capital charges for CVA risk and the trading book regime has not been clarified**: The introduction of the new capital charge for CVA risk under Basel III uses elements of the market risk framework. In fact, in the advanced approach, CVA risk is measured through the internal market risk models. This makes it advisable to consider the treatment of CVA risks in the revised market risk framework.

**Options for a new boundary to address current observed weaknesses**

No new boundary will fix all known issues with the current boundary without presenting some further difficulties. Therefore, in considering alternative options, their advantages and disadvantages need to be assessed. The Committee recognises that any disadvantages and unresolved issues identified from the ultimate choice of boundary will need to be addressed by other changes to the capital regime. This clearly includes the proposed revisions to trading book capital requirements stemming from the fundamental review. The Committee has considered a range of options for the basis of a revised trading book boundary, in addition to the removal of the boundary:

(a) Trading intent of bank management (a “trading evidence-based boundary”);

(b) Functions provided by the bank, eg market making or underwriting;

(c) Real or perceived liquidity of instruments;

(d) Risk characteristics of instruments; and

(e) The valuation methodology applied to an instrument (a “valuation-based approach”).

Boundary options based on the characteristics of instruments, or the functions provided by the bank, have conceptual merits. Nevertheless, they were considered to be too subjective to deliver a boundary that could be subject to demonstrably consistent implementation within, and across, all jurisdictions. Of the remaining three boundary options considered, the Committee felt that the benefits of considering the liquidity of instruments could be better incorporated into revised capital requirements for the trading book (rather than in the definition of the trading book itself). The Committee therefore believes that there are two approaches that are most likely to meet the described objectives whilst addressing the issues of the current boundary. These approaches are described in more detail below, and a detailed comparison is included in Annex 3. A.

**A trading evidence-based boundary**

The trading evidence-based boundary is an enhanced version of the current intent-based boundary. As such, it retains the link between the regulatory trading book and the set of instruments which a bank deems to be held for the purposes of trading (or to hedge trading book risk positions), adding more objective evidential requirements to support this principle. Fundamental to this version of the boundary is a view that a bank’s intention in holding an instrument determines the risk management strategy applied to it, and therefore is the relevant characteristic for regulators in determining its capital requirements. The proposed enhancements to the core principle of “trading intent”, the most prominent of which are set out below, are intended to provide more objective criteria for entry to the trading book and therefore make the boundary more enforceable and consistent across jurisdictions:

• As an entry requirement, instruments must be held for trading purposes (or to hedge trading book risk positions) and marked to market daily, with valuation changes recognised through the P&L account, using market data that are sufficiently robust to support this frequency of valuation.

• Banks would be required to have formal policies and documented practices for determining what instruments should be included in the trading book. This would include a description of what constitutes trading or hedging activity, and therefore what instruments should customarily be held in the trading book.

• Banks would be subject to a requirement that internal control functions conduct ongoing evaluation of instruments both in and out of the trading book, to assess whether the bank’s instruments are being properly assigned as trading or nontrading instruments in the context of the bank’s trading activities. Banks would be required to provide objective evidence that trading instruments are actively managed. This would include setting, and enforcing, limits both on an instrument and on a risk position basis. Also, in addition to clearly documented hedging strategies, banks would be required to monitor market liquidity levels (including availability of market data) and also to specify an expected maximum holding period for instruments, with potential penalties (such as required valuation adjustments/increased supervisory scrutiny) if that period is exceeded. • There would be stricter requirements on the feasibility of trading an instrument, which would supplement a requirement to have trading/hedging intent. These would include proof of access to relevant markets for trading and hedging (such as historical data on trading in those markets, or a plausible plan for how a bank would trade on a market in which it had limited experience). Banks would also need to meet minimum standards related to the periodic monitoring and assessment of the risk of trading instruments.

• If the above supervisory criteria are not met, banks would be required to designate their instruments to the banking book. At the same time, there would be a strict limit on the ability of banks to move instruments between the trading book and the banking book after initial designation at their own choice, with movement only allowed in extraordinary circumstances which would be defined in the framework. Possible examples could be a major publicly announced event, such as a bank restructuring. Many of these controls – such as the requirement for trading policies and procedures – are not new, but would be strengthened with the more detailed objective metrics to be specified. A feature of this approach is that two banks could hold the same instrument but allocate it to different books, depending upon their intention with respect to the instrument, as long as the criteria specified above are met. Thus, banks could continue to have material exposures to fair valued instruments located in the regulatory banking book that are subject only to credit risk, and not to market risk, Pillar 1 capital requirements. As such, further consideration would need to be given to whether banking book capital requirements should be adjusted to address the risk posed by such instruments.

Possible advantages and disadvantages of the trading evidence-based approach

**Advantages**

• An instrument held with trading/hedging intent, provided it is feasible that it can be freely traded or completely hedged in the short-term, appears to naturally fit into the market risk framework. The proposed changes would seek to introduce more objective conditions to improve its enforceability.

• This approach requires fewer changes to the current boundary relative to valuation-based approaches (described below), and therefore would result in less disruption to banks and supervisors upon introduction.

• The instruments within the trading book would more closely resemble the instruments held within the parts of banks that are internally described and risk managed as “trading”, as well as to trading risk metrics, which should make the framework simpler for banks to implement, and easier for supervisors with trading expertise to oversee.

**Disadvantages**

• The trading book boundary would still be under the control of banks, allowing them (restricted to some extent by the new conditions on the boundary) some flexibility to choose the designation of their instruments provided they are willing to fair value them daily through P&L and accept treatment in the trading book as long as the bank holds the position.

• There would remain a set of fair valued instruments in the banking book, which would not receive Pillar 1 market risk capital requirements.

• The consistency of the approach would rely on each jurisdiction performing sufficiently reasonable judgments on the feasibility of trading in different markets – leading to potential disparities in application across jurisdictions.

**A valuation-based boundary**

The core principle of the valuation-based boundary would move away from the concept of “trading intent” to instead construct a boundary that focuses on aligning the design and structure of regulatory capital charges with the risks posed by an instrument to the regulatory capital position of a bank. This approach would recognise the link between capital resources and capital requirements and attempt to more fully address the fact that market price changes in all instruments held at fair value immediately impact the solvency of banks. To achieve this objective, one option would be to require any fair valued balance sheet asset or liability to be subject to market risk capital charges. Strictly defined, however, this could result in a potentially large number of non-traded assets and liabilities requiring market risk capital (for example, including assets such as patents, property). A more feasible approach, which the Committee believes would avoid this complication, would be to only apply the boundary to fair-valued financial instruments. Moreover, a strict link between accounting fair value and market risk capital requirements would also potentially misalign market risk capital requirements with the instruments whose fair value movements impact capital resources under Basel III. To address this, the Committee proposes that the boundary be reduced in scope to ensure that it only covers those financial instruments where a movement in their value could lead to a reduction in capital resources under the Basel III definition of capital requirements – this aligns capital requirements with risks to capital resources. Under this approach, in current accounting terms, the new trading book would include held for trading financial instruments, available for sale financial instruments and other financial instruments to which fair value is applied either as an option or a requirement. The Committee would need to consider whether the framework’s current definition of financial instruments20 is sufficiently clear to ensure consistent enforcement. A new “trading book” under this approach would likely be significantly larger than the current trading book for many banks, increase the number of banks subject to market risk capital requirements and may differ across banks and jurisdictions due to differences in accounting standards. However, as previously discussed, this boundary would not necessarily lead to a wider scope of modelled risk positions. Potential adjustment to the valuation-based boundary: Whilst conceptually sound, the above valuation-based approach could, in some circumstances, disincentivise prudent hedging of interest rate risk in the banking book because hedges held at fair value would be split from the hedged risk position. The Committee is considering a potential adjustment to the valuation-based boundary such that banks could be permitted to include some fair valued financial instruments in the banking book if they can provide clear evidence that those financial instruments are specifically used to hedge other banking book risk positions as part of interest rate risk management arrangements. Under this option, the trading book boundary would be again partly under the control of banks, allowing them some flexibility to choose the designation of their instruments Possible advantages and disadvantages of the valuation-based approach

**Advantages**

• All financial instruments held at fair value and so subject to market risk (because changes in fair value could lead to a reduction in capital resources under the Basel III definition of capital) would be required to have market risk capital against that risk.

• The trading book boundary would more closely align with the accounting divide between instruments that are recorded at fair value, and instruments that are recorded at amortised cost. Supervisors could expend less resource monitoring the regulatory boundary, with auditors, as part of their current duties, verifying accounting classification. Some of the goals of auditors and supervisors could be better aligned.

• The default choice of whether to hold a financial instrument in the trading book or not would be largely dependent on the accounting rules and filters in the Basel III framework.24 Although the accounting rules may still leave flexibility when designating financial instruments at fair value, arbitrage opportunities are likely to be reduced.

**Disadvantages**

• The link to accounting fair value would make the trading book boundary largely dependent on decisions and changes made by accounting standard setters, and auditors’ interpretation of those standards, neither of which are under the control of the Committee.

• Jurisdictional differences in accounting, for example with regard to tainting of held to maturity securities, could result in large disparities in the scope of the trading book across banks in different countries and potentially significant increases in their regulatory trading book portfolios.

• The set of fair value financial instruments may encompass instruments that a bank does not trade. Thus, the boundary would not align with banks’ internal risk management practices for trading activities.

**Changes common to both boundary options**

Regardless of the final choice of the core principle underpinning a future boundary, there are a number of issues/improvements common to both options:

• To encourage market discipline, a set of disclosure requirements regarding the composition of the trading book would also be developed. For example, banks could be required to publish detailed information about the nature of instruments included in the trading book.

• The ability to change the designation of an instrument between trading book and banking book at the bank’s own choice would be significantly restricted either through the explicit limitation imposed by the trading evidence-based boundary or through the link to the fair value accounting requirements in the valuation-based approach.

Stronger, more specific, prudent valuation requirements would be developed and applied to all fair valued financial instruments, regardless of trading book or banking book designation.

**47. How will trading businesses change as a result of Basel III capital rules for banks’ trading books?**

The revised approach to the regulation of banks’ trading books –focusing on capturing deep losses during systemic crises and a tougher approach to internal-risk modeling – will limit lucrative arbitrage and trading opportunities.

The Basel Committee on Banking Supervision (BCBS) wants to make banks treat the assets in their trading books more like those in their banking books by forcing them to hold increased capital against assets designated for trading.

Historically, lower capital requirements for trading book assets had encouraged banks to shift assets from the more expensive banking book into the trading book. However, this left many banks with insufficient capital to cover losses when credit markets collapsed during the crisis.

However, the latest BCBS proposals for how banks should calculate risk-based capital requirements at the trading book level, published on October 31, indicate banks are winning the argument that they should continue to be allowed freedom

to use their own models to determine how much capital to hold against risky trading activities.

The revised rules represent a substantial U-turn for the BCBS, which had originally proposed in 2012to force banks to aggregate their trading desk’ risk weighted assets (RWA), according to prescribed correlations.

Fixed correlations were needed, the BCBS suggested, because different banks used different models for calculating risks, which led to an unhelpful fragmentation of oversight approaches among national regulators. All this variation meant that risks were probably being understated and capital reserves were too low.

By and large, the industry seems satisfied with the compromise, which allows banks to rely on their own internal credit risk and correlation models, but also requires them to backstop these with a standardized approach based on credit rating agency ratings.

The revised rules include new suggestions for making banks’ internal models more robust, as well as changes to the standardized approach that will increase its risk-sensitivity, while making it easier fors maller banks to implement.

“It’s tough to be entirely critical of the revised proposals, as it’s clear that some additional level of capital reserve is helpful in a market downturn,” says one market source. “The question is how much is too much.”

The BCBS is undertaking a quantitative impact study to determine how the current proposals will perform under various stressed market scenarios, and intends to report the results at the end of the consultation period in January 2014.

However, some observers argue that even in their watered down form, the new proposals show that regulators are moving beyond the objective of matching rules to market reality, and are seeking to restructure banks’ incentives for engaging in certain trading activities.

Part of the problem, they argue, is regulators are basing the new framework on the idea that a future crisis could be far worse than the one that occurred in 2008.

“A major complaint with the current focus on trading book RWA is that if you look at the experience during the crisis, liquidity didn’t dry up in the way that is being assumed here,” says Simon Gleeson, a partner with law firm Clifford Chance in London.

“However, these proposals aren’t an attempt to correlate with reality, but rather they seek to build a driver into the system – a disincentive to hold trading assets.”

Indeed, the revised trading book RWA proposal is the latest of a raft of reforms, including the liquidity coverage ratio and the credit valuation adjustment charge that seek to decouple banks’ capital reserves from the health of the asset markets they participate in.

“There’s a decent policy reason why regulators should not want banks to hold traded assets that are accounted for on a mark-to-market basis,” says Gleeson. “A market crash translates the mark-to-market moves real time into a reduction in bank capital. If a bank is holding a portfolio of non-mark-to-market assets, such as the loans in its banking book, its capital stock is not exposed.”

Given the mutually reinforcing direction of regulatory capital reforms and OTC derivatives regulation, several banks have already realigned their business models according to the new incentives, with a sharp reduction in fixed-income inventory and position-taking among leading market makers.

Jon Skinner, an independent management consultant based in New York who advises banks on their strategic response to regulatory reforms, says that Basel 2.5 already had a significant impact on market RWA levels and the trading book rules appear to be a refinement of this rather than a major further uptick in RWA.

It will be interesting to see how flow-trading dealers will evolve their market risk management and flow trading in response to the combination of market risk RWA rules (Basel 2.5, trading book review) and the Dodd-Frank SEF mandate in the US.

“Whilst prop trading is subject to well-publicized regulatory pressures, there is also market risk andhence RWA inherent in pure flow trading. Although the trading book review alone may not place afurther burden sufficient to change the trading model, given the SEF trading mandate in the US,dealers may look to optimize market risk RWA in the new environment by adjusting their flow tradingbusiness," he says.

Under the new regime, this kind of trading operation is now a dying breed.

**48. What are the key requirements of the Basel stress testing framework? Are they sufficiently stringent?**

1. Stress testing should form an integral part of the overall governance and risk management culture of the bank. Stress testing should be actionable, with the results from stress testing analyses impacting decision making at the appropriate management level, including strategic business decisions of the board and senior management. Board and senior management involvement in the stress testing programme is essential for its effective operation.

2. A bank should operate a stress testing programme that promotes risk identification and control; provides a complementary risk perspective to other risk management tools; improves capital and liquidity management; and enhances internal and external communication.

3. Stress testing programmes should take account of views from across the organisation and should cover a range of perspectives and techniques.

4. A bank should have written policies and procedures governing the stress testing programme. The operation of the programme should be appropriately documented.

5. A bank should have a suitably robust infrastructure in place, which is sufficiently flexible to accommodate different and possibly changing stress tests at an appropriate level of granularity.

6. A bank should regularly maintain and update its stress testing framework. The effectiveness of the stress testing programme, as well as the robustness of major individual components, should be assessed regularly and independently.

7. Stress tests should cover a range of risks and business areas, including at the firm-wide level. A bank should be able to integrate effectively, in a meaningful fashion, across the range of its stress testing activities to deliver a complete picture of firm-wide risk.

8. Stress testing programmes should cover a range of scenarios, including forward-looking scenarios, and aim to take into account system-wide interactions and feedback effects.

9. Stress tests should feature a range of severities, including events capable of generating the most damage whether through size of loss or through loss of reputation. A stress testing programme should also determine what scenarios could challenge the viability of the bank (reverse stress tests) and thereby uncover hidden risks and interactions among risks.

10. As part of an overall stress testing programme, a bank should aim to take account of simultaneous pressures in funding and asset markets, and the impact of a reduction in market liquidity on exposure valuation.

11. The effectiveness of risk mitigation techniques should be systematically challenged.

12. The stress testing programme should explicitly cover complex and bespoke products such as securitised exposures. Stress tests for securitised assets should consider the underlying assets, their exposure to systematic market factors, relevant contractual arrangements and embedded triggers, and Principles for sound stress testing practices and supervision 15 impact of leverage, particularly as it relates to the subordination level in the issue structure

13. The stress testing programme should cover pipeline and warehousing risks. A bank should include such exposures in its stress tests regardless of their probability of being securitized

14. A bank should enhance its stress testing methodologies to capture the effect of reputational risk. The bank should integrate risks arising from off-balance sheet vehicles and other related entities in its stress testing programme.

15. A bank should enhance its stress testing approaches for highly leveraged counterparties in considering its vulnerability to specific asset categories or market movements and in assessing potential wrong-way risk related to risk mitigating techniques.

**49. Which extreme events should stress tests be taking into consideration now?**

The three key areas stress tests focus on the most are credit risk, market risk, and liquidity risk

That guidance identified eight risk categories including credit risk, country and transfer risk, market risk, interest rate risk, liquidity risk, operational risk, legal risk and reputational risk.

**Ordinary Times**

Credit risk Defaults may be predicted based on historical data—no strong correlation of defaults

Market risk Loss of value linked to performance of asset

Operation risk More likely to be institution specific

**Extreme Events**

Credit risk Inadequate data for robust quantification or risk— strong correlation of default

Market risk Loss of value linked to type of asset (contagion)

Operation risk More likely to be environmental

Extreme events can be usefully divided into two categories. The fi rst includes identifi able outcomes (often predicted by history) that occur too infrequently to be effectively managed. Think of the fi nancial equivalents of a 1,000-year fl ood, an 8.5 point earthquake centered on Los Angeles or a meteor strike. Hyperinfl ation and systemic bank failures have occurred—we simply feel these events are extremely unlikely to occur at any given moment in time, and their impacts are likely to be so overwhelming that risk mitigation does not appear to be worth the effort.5 At the outer limits, hopelessness sets in. The second category of extreme events is more frightening. These are the inevitably unaddressed risks associated with unidentifi ed events—events for which there is no historical experience. Taleb teaches (among other things) that novel events do occur. Our imagination, and hence our ability to anticipate outcomes, is limited by our experiences. Our habits, however, instruct us to assume continuity

Warehousing/Pipeline Risk

The current fi nancial crisis brought home a category of underappreciated risk. Banks engaged in active originate-and-distribute activity anticipated very short holding periods for bank-originated assets. These assets were either sold in secondary markets or packaged into securitization vehicles. Warehousing and pipeline risk refers to the event where originating banks are unable to off-load assets due to unexpected changes in market conditions. Involuntary holding of these assets expose the bank to losses due to declining values of these assets.

Reputation Risk

Reputation risk refers both to the prospect of a decline in value of a bank’s goodwill, as well as the possibility that a bank will feel constrained to undertake certain transactions in order to maintain goodwill. This particular risk manifested during the current crisis. Banks involved in securitization activities frequently transferred assets to formally independent (and bankruptcy remote) corporate entities (often called Special Purpose Entities). Once these assets were transferred, they no longer appear on the bank’s balance sheet. Nor were they treated as off-balance sheet items, unless the bank was contractually obligated to intervene in event of default or other stress. Many banks re-acquired distressed assets from Special Purpose Entities, notwithstanding the absence of a contractual obligation to do so. Recognition of this moral obligation is explained by concerns by banks for their reputations. Banks realized avoidable fi nancial losses in order to avoid investor wrath. Given this history, it is likely that implicit puts will be recognized when securitization markets eventually re-emerge. Extreme Operational Risks Operational risk describes the residual category of identifi able risks beyond credit and markets risks. These include both internal and environmental risks. Internal risks include poor management, losses caused by rogue traders, and fraud. It also includes certain counterparty risks that are not captured under the credit or market risk categories. External risks include those presented by weaknesses in the interlocked fi nancial system, as well as the usual horrors (wars, plagues, insurrections). An operational risk event may be of an unanticipated magnitude— and not adequately provided for. And it may be perversely correlated with other stresses. Consider the Madoff Ponzi scheme. A Ponzi scheme is subject to collapse even under the most benign conditions, but is much more vulnerable during periods of market pessimism and declining asset values. The recognition of losses from such an event (corresponding to discovery and collapse of the scheme) may well compound simultaneous credit and market losses

During extreme events salient operational risks are more likely to be external—that is, they cannot be so easily traced to management failures. And so they are more likely be experienced by many institutions—leading in turn to magnifi cation and systemic risk. Not all risks can be anticipated. Indeed, it is diffi cult to manage risks that cannot be imagined (those that are outside of experience) although experience teaches that unanticipated and unimagined events do occur. The liquidity crisis had been imagined by some, who—like Cassandra—were ignored. That said, one might credibly say that the “market” failed to imagine or anticipate such an outcome.

Extreme Systemic Risk

Basel II directly controls the activities of individual institutions. That said, like all prudential regulation, Basel II is in some sense more concerned by the spillover effects of an institution’s crisis onto the larger banking system.10 In the current fi nancial crisis, bank capital was not only inadequate as measured on an individual institutional basis; it was inadequate on a system-wide basis. There were no suffi ciently well capitalized banks that were able to absorb the capital defi cits (and negative equity) of the failed institutions. Government response in the United States, United Kingdom and elsewhere demonstrated the existence of severe de facto capital defi cits. While institutions remain with well-worn, recognizable names, the effective truth is that the pre-crisis banking sector has been utterly destroyed and what exists today results from an ad hoc public recapitalization. Schwarcz has argued that a better approach to the problem of systemic risk should be located outside particular banking institutions in a “liquidity provider of last resort.” In some sense, Schwarcz anticipated the immediate crisis’ responses, which might be described as a bundle of state-provided liquidity and capital infusions to the broad banking sector.

**50. Why is Basel II blamed for precipitating the 2008 financial crisis?**

Among the things that caused the financial crisis was that the Basel II committee and banks underestimated both the risk of losses on their assets and their exposure to the failure of others

One weakness of Basel II emerged during the subprime mortgage meltdown and Great Recession of 2008 when it became clear that Basel II underestimated the risks involved in current banking practices and that the financial system was overleveraged and undercapitalized.

Basel II’s failure, I argue, lies in regulatory capture, ‘de facto control of the state and its regulatory agencies by the ‘regulated’ interests, enabling these interests to transfer wealth to themselves at the expense of society’.3 Large international banks were able to systematically manipulate outcomes in Basel II’s regulatory process to their advantage, at the expense of their smaller and emerging market competitors and, above all, systemic financial stability. To understand why this happened, I present an analytical framework which sets out the broad conditions under which capture is expected to occur. My framework draws on what I call the ‘neo-proceduralist’ school of global regulation, developed in recent work by Walter Mattli and Ngaire Woods, which emphasizes two types of conditions. The first are so-called ‘supply-side’ conditions concerning the institutional context in which Basel II was drafted, and the second are ‘demand-side’ conditions concerning the extent of societal pressure for new regulation. I argue, however, that the neo-proceduralism can be strengthened as a theory of global regulatory processes by proper temporal contextualization. It is only by conceiving of capture as a process that unfolds over time that we can appreciate exactly how supply- and demand-side factors combined to give large international banks disproportionate influence over the Basel process. As it will later become clear, this theoretical innovation has implications that go beyond Basel II. It allows us to understand not only why the Basel Committee failed to achieve its objectives for the accord, but also why some of the more latest proposals in international banking regulation – despite the tremendous political will behind them – have enjoyed no more success. The failure of these proposals, my analysis warns, is very much a case of history repeating itself.

# ****Other Quant interview questions****

## Options

1. **Have a working knowledge of options and options pricing/risk factors.**

Options can be used in a wide variety of strategies, from conservative to high risk. They can also be tailored to meet expectations that go beyond simple directional strategies. So, once you learn basic options terminology, it makes sense to investigate factors that affect an option's price in various scenarios.

**KEY TAKEAWAYS**

Options are derivative contracts the right, but not the obligation, to buy (for a call option) or sell (for a put option) some asset at a pre-determined price on or before the contract expires.

Options can be used for directional strategies or to hedge against certain risks in the market.

Pricing an option relies on complex mathematical formulas, but the direct inputs into an option's price include the price of the underlying asset, the option's strike, time to expiration, interest rates, and implied volatility.

**Using Options For Directional Strategies**

When stock traders first begin using options, it is usually to purchase a call or a put for directional trading, in which they expect a stock will move in a particular direction. These traders may choose an option rather than the underlying stock due to limited risk, high reward potential, and less capital required to control the same number of shares.

If the outlook is positive (bullish), buying a call option creates the opportunity to share in the upside potential without having to risk more than a fraction of the market value. If bearish, buying a put lets the trader take advantage of a fall without the margin required to sell short.

**Market Direction And Value**

Many kinds of option strategies can be constructed but the position's success or failure depends on a thorough understanding of the two types of options: the put and the call. Furthermore, taking full advantage of options requires a new way of thinking because traders who think solely in terms of market direction miss all sorts of opportunities.

In addition to moving up or down, stocks can move sideways or trend modestly higher or lower for long periods of time. They can also make substantial moves up or down in price, then reverse direction and wind up back where they started. These kinds of price movements cause headaches for stock traders but give option traders the exclusive opportunity to make money even if the stock goes nowhere. Calendar spreads, straddles, strangles and butterflies highlight a few option strategies designed to profit in those types of situations.

**Basics of Option Pricing**

Option traders need to understand additional variables that affect an option's price and the complexity of choosing the right strategy. Once a stock trader becomes good at predicting the future price movement. They may believe it is an easy transition from options but this isn't true. Options traders must deal with three shifting parameters that affect the price: the price of the underlying security, time, and volatility. Changes in any or all of these variables affect the option's value.

Option pricing theory uses variables (stock price, exercise price, volatility, interest rate, time to expiration) to theoretically value an option. Essentially, it provides an estimation of an option's fair value which traders incorporate into their strategies to maximize profits. Some commonly used models to value options are Black-Scholes, binomial option pricing, and Monte-Carlo simulation. These theories have wide margins for error due to deriving their values from other assets, usually the price of a company's common stock. There are mathematical formulas designed to compute the fair value of an option. The trader simply inputs known variables and gets an answer that describes what the option should be worth.

The primary goal of any option pricing model is to calculate the probability that an option will be exercised, or be in-the-money (ITM), at expiration. Underlying asset price (stock price), exercise price, volatility, interest rate, and time to expiration, which is the number of days between the calculation date and the option's exercise date, are commonly used variables that are input into mathematical models to derive an option's theoretical fair value.

**Key Pricing Inputs**

Here are the general effects that variables have on an option's price:

**1. Underlying Price & Strike Price**

The value of calls and puts are affected by changes in the underlying stock price in a relatively straightforward manner. When the stock price goes up, calls should gain in value because you are able to buy the underlying asset at a lower price than where the market is, and puts should decrease. Likewise, put options should increase in value and calls should drop as the stock price falls, as the put holder gives the right to sell stock at prices above the falling market price.

That pre-determined price at which to buy or sell is called the option's strike price or exercise price. If the strike price allows you to buy or sell the underlying at a level which allows for an immediate profit buy disposing of that transaction in the open market, the option is in-the-money (for example a call to buy shares at $10 when the market price is currently $15, you can make an immediate $5 profit).

**2. Time to Expiration**

The effect of time is easy to conceptualize but takes experience before understanding its impact due to the expiration date. Time works in the stock trader's favor because good companies tend to rise over long periods of time. But time is the enemy of the buyer of the option because, if days pass without a significant change in the price of the underlying, the value of the option will decline. In addition, the value of an option will decline more rapidly as it approaches the expiration date. Conversely, that is good news for the option seller, who tries to benefit from time decay, especially during the final month when it occurs most rapidly.

**3. Interest Rates**

Like most other financial assets, options prices are influenced by prevailing interest rates, and are impacted by interest rate changes. Call option and put option premiums are impacted inversely as interest rates change: calls benefit from rising rates while puts lose value. The opposite is true when interest rates fall.

**4. Volatility**

The effect of volatility on an option's price is the hardest concept for beginners to understand. It relies on a measure called statistical (sometimes called historical) volatility, or SV for short, looking at past price movements of the stock over a given period of time.

Option pricing models require the trader to enter future volatility during the life of the option. Naturally, option traders don't really know what it will be and have to guess by working the pricing model "backwards". After all, the trader already knows the price at which the option is trading and can examine other variables including interest rates, dividends, and time left with a bit of research. As a result, the only missing number will be future volatility, which can be estimated from other inputs

These inputs form the core of implied volatility, a key measure used by option traders. It is called implied volatility (IV) because it allows traders to determine what they think future volatility is likely to be.

Traders use IV to gauge if options are cheap or expensive. You may hear option traders say that premium levels are high or that premium levels are low. What they really mean is that the current IV is high or low. Once understood, the trader can determine when it is a good time to buy options - because premiums are cheap - and when it is a good time to sell options - because they are expensive.

**The Bottom Line**

Options are complex, but their price can be described by just a handful of variables, most of which are known in advance. Only the volatility of the underlying asset remains a matter of estimation. Once you have a firm grasp of the essentials, you'll find that options provide flexibility to tailor the risk and reward of every trade to your individual strategies.

1. **Understand**[**delta**](https://www.wallstreetoasis.com/finance-dictionary/what-is-delta)[**hedging**](https://www.wallstreetoasis.com/resources/skills/trading-investing/hedging)**.**

One of the building blocks of derivatives theory is delta hedging. This is the theoretically perfect elimination of all risk by using a very clever hedge between the option and its underlying.

Delta hedging exploits the perfect correlation between the changes in the option value and the changes in the stock price. This is an example of ‘dynamic’ hedging; the hedge must be continually monitored and frequently adjusted by the sale or purchase of the underlying asset. Because of the frequent rehedging, any dynamic hedging strategy is going to result in losses due to transaction costs. In some markets this can be very important. The ‘underlying’ in a delta-hedged portfolio could be a traded asset, a stock for example, or it could be another random quantity that determines a price such as a risk of default. If you have two instruments depending on the same risk of default, you can calculate the sensitivities, the deltas, of their prices to this quantity and then buy the two instruments in amounts inversely proportional to these deltas (one long, one short). This is also delta hedging. If two underlyings are very highly correlated you can use one as a proxy for the other for hedging purposes. You would then only be exposed to basis risk. Be careful with this because there may be times when the close relationship breaks down. If you have many financial instruments that are uncorrelated with each other then you can construct a portfolio with much less risk than any one of the instruments individually. With a large such portfolio you can theoretically reduce risk to negligible levels. Although this isn’t strictly hedging it achieves the same goal.

1. **Given two options with equal strike price on the same underlying, one with an exercise date in a month, the other in three months, which is more valuable?**

The one three months away (more time to expiration -> more likelihood of being in the money). Unless the option is deep in the money -> the shorter one should be worth more

The further out an options contract is, the more expensive the premium, because there’s more likelihood the option will expire ITM.

So as time goes on, and the expiry date gets closer, the value of the option will decrease.

Theta is the change of option price with respect to time. It’s negative because the option gets cheaper as time goes forward

E.g. call option -> stock price is USD 20, strike price is USD 19, Volatility is 15%, risk free rate is 1%.

If the maturity is 10 months away, the option is worth quite a bit of money.

If the maturity is 10 seconds away, the option is worthless.

The fact that the option looses value over time (all else equal) is the reason that Theta is generally negative.

## Fixed Income

1. **Know bond math and duration and convexity.**

Duration and convexity are statistics that estimate the sensitivity of the market value of an asset or liability to a change in interest rates. Usually the asset or liability is a fixed-income bond, but as measures of rate sensitivity, they apply to all sorts of securities and derivatives. We can ask meaningfully about the duration and convexity of a floating-rate note, an inflation-indexed bond, or an interest rate swap. That discussion will have to wait until Chapters 7 and 8. This chapter focuses on the risk statistics applicable to a typical fixed-rate or zero-coupon bond.

We start with classic yield duration – the sensitivity of the bond price to a change in its yield to maturity. This leads to the well-known Macaulay and modified duration statistics. Yield convexity is the second-order effect of that yield change. The beauty of yield duration and convexity is that they are based on fundamental mathematical properties of the bond. That means closed-form formulas can be derived for the statistics using algebra and calculus. Then we move on to other descriptions of change in interest rates – the sensitivity of the bond price to a shift in the benchmark Treasury yield curve. I call these curve duration and curve convexity.

Before diving into the bond math, let's get a sense of interest rate sensitivity using an admittedly contrived scenario. Suppose that you are the fixed-income strategist for an aggressively managed, high-yield, international bond fund. You believe that the market prices of some country's long-term bonds will rally in the next week as the market digests what you expect to be very positive news about economic conditions. In particular, two long-term sovereign bonds are trading at deeply discounted prices to yield 20%. Both bonds have an annual coupon rate of 6%, paid once a year. One bond matures in 20 years and the other in 30 years; otherwise they are identical. Which bond do you recommend, assuming that you anticipate both yields to drop by 100 basis points from 20% to 19% – the 20-year or the 30-year bond?

I've posed this problem over the last 25 years to hundreds of students, including some emerging-market, fixed-income traders. Virtually all recommend the 30-year bond; you probably do, too. Your thinking probably is that, other things being equal (meaning the coupon rate, yield to maturity, payment frequency, default risk, liquidity, taxation), the longer-term 30-year bond gains or loses more value on a percentage basis than the shorter-term 20-year bond given the same change in yield. That's true – almost all of the time. We see later in the chapter that this intuition is not always correct, and in this case, “almost always” needs to be entered into the statement. This scenario of two high-yield, deeply discounted bonds demonstrates a real bond math curiosity – the 30-year bond actually has lower sensitivity to a change in its yield to maturity than the 20-year bond. To understand this oddity, you need to explore the mathematics behind duration.

1. **You might get technicals on these things. Which bond has more**[**interest rate**](https://www.wallstreetoasis.com/resources/skills/finance/interest-rate)**risk, a 3yr zero or a 3yr coupon?**

Zero -> higher duration

1. **Which bond has more interest rate risk a 3yr zero a 5yr zero?**

5YR -> higher duration

1. **Suppose a desk buys a corporate bond from a client but wants to get risk-flat. Which risks does it need to hedge?**

* **Interest rate risk**

**Forwards**: A forward contract is the most basic interest rate management product. The idea is simple, and many other products discussed in this article are based on this idea of an agreement today for an exchange of something at a specific future date.

**Forward Rate Agreements (FRAs):** An FRA is based on the idea of a forward contract, where the determinant of gain or loss is an interest rate. Under this agreement, one party pays a fixed interest rate and receives a floating interest rate equal to a reference rate. The actual payments are calculated based on a notional principal amount and paid at intervals determined by the parties. Only a net payment is made—the loser pays the winner, so to speak. FRAs are always settled in cash.

FRA users are typically borrowers or lenders with a single future date on which they are exposed to interest rate risk. A series of FRAs is similar to a swap (discussed below); however, in a swap, all payments are at the same rate. Each FRA in a series is priced at a different rate unless the term structure is flat.

**Futures**: A futures contract is similar to a forward, but it provides the counterparties with less risk than a forward contract—namely, a lessening of default and liquidity risk due to the inclusion of an intermediary.

**Swaps**: Just like it sounds, a swap is an exchange. More specifically, an interest rate swap looks a lot like a combination of FRAs and involves an agreement between counterparties to exchange sets of future cash flows. The most common type of interest rate swap is a plain vanilla swap, which involves one party paying a fixed interest rate and receiving a floating rate, and the other party paying a floating rate and receiving a fixed rate.

Diversification is one method to hedge against interest rate risk.

**Options**: Interest rate management options are option contracts for which the underlying security is a debt obligation. These instruments are useful in protecting the parties involved in a floating-rate loan, such as adjustable-rate mortgages (ARMs). A grouping of interest rate call options is referred to as an interest rate cap; a combination of interest rate put options is referred to as an interest rate floor. In general, a cap is like a call, and a floor is like a put.

**Swaptions**: A swaption, or swap option, is simply an option to enter into a swap.

**Embedded Options**: Many investors encounter interest management derivative instruments via embedded options. If you have ever bought a bond with a call provision, you too are in the club. The issuer of your callable bond is insuring that if interest rates decline, they can call in your bond and issue new bonds with a lower coupon.

**Caps**: A cap, also called a ceiling, is a call option on an interest rate. An example of its application would be a borrower going long, or paying a premium to buy a cap and receiving cash payments from the cap seller (the short) when the reference interest rate exceeds the cap's strike rate. The payments are designed to offset interest rate increases on a floating-rate loan.

If the actual interest rate exceeds the strike rate, the seller pays the difference between the strike and the interest rate multiplied by the notional principal. This option will "cap," or place an upper limit, on the holder's interest expense.

The interest rate cap is a series of component options, or "caplets," for each period the cap agreement exists. A caplet is designed to provide a hedge against a rise in the benchmark interest rate, such as the London Interbank Offered Rate (LIBOR), for a stated period.

**Floors**: Just as a put option is considered the mirror image of a call option, the floor is the mirror image of the cap. The interest rate floor, like the cap, is a series of component options, except that they are put options and the series components are referred to as "floorlets." Whoever is long, the floor is paid upon maturity of the floorlets if the reference rate is below the floor's strike price. A lender uses this to protect against falling rates on an outstanding floating-rate loan.

**Collars**: A protective collar can also help manage interest rate risk. Collaring is accomplished by simultaneously buying a cap and selling a floor (or vice versa), just like a collar protects an investor who is long on a stock. A zero-cost collar can also be established to lower the cost of hedging, but this lessens the potential profit that would be enjoyed by an interest rate movement in your favor as you have placed a ceiling on your potential profit.

* **Reinvestment risk**

Investors can reduce reinvestment risk by holding bonds of different durations and by hedging their investments with interest rate derivatives. Having a fund manager can help reduce reinvestment risk; therefore, some investors consider allocating money into actively managed bond funds.

* **Credit / counterparty risk**

Credit Default Swaps

1. **How might it go about hedging those risks?**

As in point 4

1. **How would I calculate the duration of a portfolio of bonds from the individual durations?**

There are two methods for calculating the duration of a bond portfolio:

* the weighted average of time to receipt of the aggregate cash flows; and.
* the weighted average of the individual bond durations comprising the portfolio.

## Volatility

1. **What is the volatility surface?**

The widespread practice of quoting option prices in terms of their Black-Scholes implied volatilities(IVs) in no way implies that market participants believe underlying returns to be lognormal. On the contrary, the variation of IVs across option strike and term to maturity, which is widely referred to as the volatility surface, can be substantial. In this brief review, we highlight some empirical observations that are most relevant for the construction and validation of realistic models of the volatility surface for equity indices. The Shape of the Volatility Surface. Ever since the 1987 stock market crash, volatility surfaces for global indices have been characterized by the volatility skew: For a given expiration date, implied volatilities increase as strike price decreases for strikes below the current stock price (spot) or current forward price. This tendency can be seen clearly in the S&P500 volatility surface shown in Figure 1. For short-dated expirations, the cross-section of IVs as a function of strike is roughly Vshaped, but has a rounded vertex and is slightly tilted. Generally this V-shape softens and becomes flatter for longer dated expirations, but the vertex itself may rise or fall depending on whether the term structure of ATM volatility is upward or downward sloping. Conventional explanations for the volatility skew include:

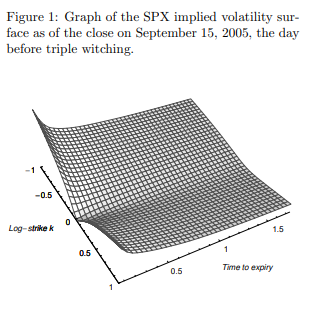
• *The leverage effect*: Stocks tend to be more volatile at lower prices than at higher prices.

• *Volatility* moves and spot moves are anticorrelated.

• *Big jumps* in spot tend to be downwards rather than upwards.

• *The risk of default*: There is a nonzero probability for the price of a stock to collapse if the issuer defaults.

• *Supply and demand*: Investors are net long of stock and so tend to be net buyers of downside puts and sellers of upside calls.

****

1. **How do you find it/where does it come from?**

**Thinking about Building a Volatility Surface? Think Again**

At first glance, constructing a volatility surface looks like a straightforward exercise – identify options that trade on the assets or securities of interest, obtain prices for those options across strikes and expirations, and compute implied vols from those prices. Voila. But that description only scratches the surface (pardon the pun) – a closer look reveals there is a great deal more to consider. In fact, constructing a thoughtful, reliable volatility surface (and updating it every day), may be one of the more challenging, painstaking aspects of maintaining a robust risk management system.

In this article, we summarize the process and validation procedures TS Imagine uses to construct implied volatility surfaces for equity, equity index and commodity options. This multi-step process includes choices of option models and price snaps, chain reconciliation, volatility calculation, and smoothing. We provide a high-level description of these issues and include some technical details for those clients who are interested.

In addition to the standard Black Scholes model used to price European options, the following models, available in TS Imagine, are commonly used in the industry:

**Finite difference** – Discretizes the continuous approach to solving differential equations, approximating them with difference equations and using finite differences to calculate the derivatives.

**Baroni-Adesi and Whaley** – To price American options, this model separately computes the European option value, then estimates the early exercise premium using a quadratic equation.

**Full Binomial / Discrete Yield Binomial** – Binomial trees model the evolution of underlying option variables over time. While computationally more intensive than Black Scholes, this approach can be more accurate, especially for long-dated options on securities with dividend payouts, which can be treated as discrete yields.

**Which Option Models?**

The first step is to choose the option models to use when calculating implied volatility from a given price. Different models produce different answers from the same inputs, so it is important to select the model that is best suited for each instrument. TS Imagine uses a Finite Difference model for American options and the Black Scholes model for European options. Other models, specifically Baroni-Adesi and Whaley, and Full Binomial and Discrete Yield Binomial models, are also available so that clients can customize their vol surfaces using the model(s) of their choice.

**Price? Which Price?**

Once you have chosen the option model to use, calculating the implied volatility of the option is a mathematical exercise – but what price do you use? Many (perhaps most) options markets are rather illiquid. Should one use the last traded price? That may sound appealing since it reflects an actual transaction, but the last trade could have occurred many hours or even days prior to the close of the market for the underlying. Our preference is to use the Settlement Price from the Exchange, but if none is available it is reasonable to use the bid, mid or ask price as of the close. However, that requires another decision; using one or the other will produce different answers. We use the mid-price, as a compromise to “minimize” the difference.

Of course, when constructing a volatility surface it is important that the prices used be as timely and current as possible, but not all prices are available at the same time. The market for the underlying and the market where the options are traded might close at different times. TS Imagine has checks in place to make sure both the underlyings and option prices are available and are “snapped” at the same time. Of course, there are time zone differences to consider also. To cover markets globally, we maintain three regions, East, West and GMT. If data for the East region is complete, that vol surface is refreshed without waiting for markets in the other regions to close. If a portfolio has a mix of securities that cut across time zones, it may be appropriate to wait until all of the volatility curves and surfaces are refreshed for a given day, but that may not be optimal; running reports earlier on some portion of the portfolio may be preferable.

**Price Filtering**

After deciding which option models and prices to use, and establishing processes to capture prices that are consistent between the options and the underlying consistently, can we declare victory and publish the vol surface? Not yet. It is critical to establish reasonableness checks and filter out prices that fail to meet those criteria. For example, if a series of option prices are not monotonically increasing or decreasing across maturities or strikes, the resulting volatility surface won’t make sense and could suggest there is a risk-free arbitrage opportunity that does not truly exist. Similarly, mixing option prices from different days will produce a bad curve for a given underlying. This may not be obvious to the end-user by visual inspection, but it communicates bad information and there must be a process in place to prevent that from happening.

TS Imagine performs the following filters on all equity and commodity options to decide whether or not they will be included in modeling the volatility surface.

Eliminate all options for which prices have not been snapped over the last five business days. To establish our filtering criteria, we individually examined over 30 exchanges representing over 3-4 million option contracts to determine the quality of the theoretical, evaluated prices each exchange would send in the absence of a traded price. Excluding all illiquid option prices is not the best decision, as it would eliminate useful info. For example, while the EUREX option market is illiquid, if the option is not traded on a given day the exchange can still send a useful settlement price. In cases where the exchange does not provide a Settlement price, such as in the U.S., we generally use the mid-price.

Eliminate in the money options. This means the volatility surface focuses on the time value of the options, rather than on any intrinsic value. The rule seems simple enough, but again there is more to it than meets the eye. How does one determine the at-the-money strike? If that line is drawn in the wrong place, the result is a different (and probably bad) volatility surface. At TS Imagine, we use listed futures prices to decide where to draw an at-the-money line, relying on contracts that are expiring on or close to the date the option expires to make the determination (note that the choice of bid/ask/mid also matters in this decision).

In cases where there are no usable prices for out-of-the-money options, or the maturity date of the longest futures contract is shorter than the furthest option expiry date so that we cannot use the futures market to determine the at-the-money strike, we can price a synthetic forward contract for that strike level, or for a maturity date corresponding to that option expiry and use put-call parity to compute prices. If there are no futures contracts on the underlying, the point at which put options become more expensive than call options can indicate the region of the ATM strike, and we can use put-call parity to determine a precise ATM line. This approach can have a tremendous impact on the integrity of the volatility surface. Since there are no futures on individual stocks, although using dividend forecasts is an acceptable method of projecting forward prices under some conditions, we believe a better approach is to use dividend futures when such a market exists. For stocks that do not pay dividends, using put-call parity to imply forward prices is preferable.

Eliminate options where prices are not increasing or decreasing monotonically. For each expiry, we sort puts and calls by strike and find the longest non-decreasing/non-increasing price sub-sequence. We select strikes from the longest sub-sequence and discard the rest.

Eliminate options where the bid/ask spread is too high. For each expiry, we calculate the mean and standard deviation of the spread and eliminate options with spreads greater than four standard deviations higher than the mean. This weeds out bad prices that can creep in for various reasons.

Use only those options that should be included in a given vol surface. For example, options on the same underlying that have the same expiry dates but different expiry frequencies (e.g., monthly versus quarterly) have different degrees of liquidity and therefore different prices. TS Imagine gives clients the flexibility to use weekly vol surfaces to price weekly options, and so on. We also have a separate futures options curve for contracts such as the S&P 500, rather than simply using the spot S&P 500 index option curve.

Check for zeroes – If settlement price is used, eliminate those with a price of zero; if the mid-price is used, eliminate contracts with a bid and/or ask equal to zero.

Remove options that have been identified as illiquid from selected exchanges (we have filtering criteria in place for each exchange that can be easily modified to apply exceptions as needed).

After completing the above steps, we check the remaining strikes for a given expiry. If the number of remaining strikes is below 20% of the median number of strikes on the curve, we exclude that expiry from the volatility surface.

Note that all of these filtering decisions were made in consultation with our clients.

**Liquidity Checks**

Lastly, we check for liquidity. We consider an option to be illiquid if one of the following conditions is met:

The option has not been traded for last five business days.

The option has no volume or open interest and the Last Trade date (according to Refinitiv) is N/A.

Some exchanges calculate theoretical settlement prices for illiquid options. In such cases, if the theoretical price improves the volatility surface, that contract or exchange can be excluded from the above checks. Further, in some markets (such as the U.S. equity options market), bid and ask prices are good enough to create a volatility surface even though the options may not be very liquid, so those exchanges can be excluded from the above checks as well. Whether a given underlying or exchange should be excluded from these price check criteria requires judgment. TS Imagine’s Data Team investigates and decides these exceptions on a case-by-case basis.

1. **What is the volatility smile?**

Volatility is annualized standard deviation of returns. Or is it? Because that is a statistical measure, necessarily backward looking, and because volatility seems to vary, and we want to know what it will be in the future, and because people have different views on what volatility will be in the future, things are not that simple.

Volatility smile is a phenomenon based on the fact that implied volatility from option prices for different strikes is not constant but instead is shaped as “smile” (deep out- and in-the money options are more volatile than at-the-money) or “skew”.

1. **Why does it exist?**

Several hypotheses explain the existence of volatility smiles.

* The simplest and most obvious explanation is that **demand** is greater for options that are in-the-money or out-of-the-money as opposed to at-the-money options.
* Others suggest that **better-developed options models** have led to out-of-the-money options becoming priced **more expensively to account for risk of extreme market crashes or black swans**. This calls into question any investing strategy that relies too heavily on implied volatility from the Black-Scholes model, particularly with the valuation of downside puts that are far away from the money.

## VaR

1. **What is VaR?**

Value-at-Risk – it is a risk measure, which returns the maximum loss generated by a portfolio in a given time period, give a predefined confidence level (with a predefined probability). It does not explain what is the maximum loss possible or in general what is the distribution of losses beyond the confidence level.

1. **How do you interpret it?**

For example, if the 95% one-month VAR is $1 million, there is 95% confidence that over the next month the portfolio will not lose more than $1 million.

1. **What are the shortcomings of VaR?**

* Lack of subadditivity of risks, even independent risks, which creates aggregation problems (i.e., VaR of a portfolio with two instrument may be greater than the sum of the independent VaRs of these instruments).
* VaR does not encourage and, indeed, sometimes prohibits diversification.

## Portfolio management

1. **How do you find the variance of a portfolio of securities?**

*Portfolio variance = w12σ12 + w22σ22 + 2w1w2Cov1,2*

1. **What information do you need?**

Weights, variances, covariances

1. **If I have an equally weighted portfolio of two securities, one with a standard deviation of returns of 2, the other is 4, what is the standard deviation of the portfolio?**

*Portfolio variance = 0.5^2 \* 2^2 + 0.5^2 \* 4 ^ 2 + 2 \* 0.5 \* 0.5 \*cov(1,2) = 5 (if cov(1,2) = 0)*

1. **Do you need more information to calculate?**

*cov(1,2)*

1. **What are different measures of "risk" of a portfolio**

Standard deviation, VaR, ES, SVaR, Max Loss, Incremental VaR, MVaR

* + Value at risk (VaR) is the minimum loss in either currency units or as a percentage of portfolio value that would be expected to be incurred a certain percentage of the time over a certain period of time given assumed market conditions.
  + VaR requires the decomposition of portfolio performance into risk factors
  + The three methods of estimating VaR are the **parametric method, the historical simulation method, and the Monte Carlo simulation method.**
  + **The parametric method of VaR** estimation typically provides a VaR estimate from the left tail of a normal distribution, incorporating the expected returns, variances, and covariances of the components of the portfolio.
  + The parametric method exploits the simplicity of the normal distribution but provides a poor estimate of VaR when returns are not normally distributed, as might occur when a portfolio contains options.
  + The historical simulation method of VaR estimation uses historical return data on the portfolio’s current holdings and allocation.
  + The historical simulation method has the advantage of incorporating events that actually occurred and does not require the specification of a distribution or the estimation of parameters, but it is only useful to the extent that the future resembles the past.
  + The Monte Carlo simulation method of VaR estimation requires the specification of a statistical distribution of returns and the generation of random outcomes from that distribution.
  + The Monte Carlo simulation method is extremely flexible but can be complex and time consuming to use.
  + There is no single right way to estimate VaR.
  + The advantages of VaR include the following: It is a simple concept; it is relatively easy to understand and easily communicated, capturing much information in a single number. It can be useful in comparing risks across asset classes, portfolios, and trading units and, as such, facilitates capital allocation decisions. It can be used for performance evaluation and can be verified by using backtesting. It is widely accepted by regulators.
  + The primary limitations of VaR are that it is a subjective measure and highly sensitive to numerous discretionary choices made in the course of computation. It can underestimate the frequency of extreme events. It fails to account for the lack of liquidity and is sensitive to correlation risk. It is vulnerable to trending or volatility regimes and is often misunderstood as a worst-case scenario. It can oversimplify the picture of risk and focuses heavily on the left tail.
  + There are numerous variations and extensions of VaR, including conditional VaR (CVaR), incremental VaR (IVaR), and marginal VaR (MVaR), that can provide additional useful information.
  + Conditional VaR is the average loss conditional on exceeding the VaR cutoff.
  + Incremental VaR measures the change in portfolio VaR as a result of adding or deleting a position from the portfolio or if a position size is changed relative to the remaining positions.
  + MVaR measures the change in portfolio VaR given a small change in the portfolio position. In a diversified portfolio, MVaRs can be summed to determine the contribution of each asset to the overall VaR.
  + **Ex ante tracking error** measures the degree to which the performance of a given investment portfolio might deviate from its benchmark.
  + Sensitivity measures quantify how a security or portfolio will react if a single risk factor changes. Common sensitivity measures are beta for equities; duration and convexity for bonds; and delta, gamma, and vega for options. Sensitivity measures do not indicate which portfolio has greater loss potential.
  + Risk managers can use deltas, gammas, vegas, durations, convexities, and betas to get a comprehensive picture of the sensitivity of the entire portfolio.
  + Stress tests apply extreme negative stress to a particular portfolio exposure.
  + Scenario measures, including stress tests, are risk models that evaluate how a portfolio will perform under certain high-stress market conditions.
  + Scenario measures can be based on actual historical scenarios or on hypothetical scenarios.
  + Historical scenarios are scenarios that measure the portfolio return that would result from a repeat of a particular period of financial market history.
  + Hypothetical scenarios model the impact of extreme movements and co-movements in different markets that have not previously occurred.
  + Reverse stress testing is the process of stressing the portfolio’s most significant exposures.
  + Sensitivity and scenario risk measures can complement VaR. They do not need to rely on history, and scenarios can be designed to overcome an assumption of normal distributions.
  + Limitations of scenario measures include the following: Historical scenarios are unlikely to re-occur in exactly the same way. Hypothetical scenarios may incorrectly specify how assets will co-move and thus may get the magnitude of movements wrong. And, it is difficult to establish appropriate limits on a scenario analysis or stress test.
  + Constraints are widely used in risk management in the form of risk budgets, position limits, scenario limits, stop-loss limits, and capital allocation
  + Risk budgeting is the allocation of the total risk appetite across sub-portfolios.
  + A scenario limit is a limit on the estimated loss for a given scenario, which, if exceeded, would require corrective action in the portfolio.
  + A stop-loss limit either requires a reduction in the size of a portfolio or its complete liquidation (when a loss of a particular size occurs in a specified period).
  + Position limits are limits on the market value of any given investment.
  + Risk measurements and constraints in and of themselves are not restrictive or unrestrictive; it is the limits placed on the measures that drive action.
  + The degree of leverage, the mix of risk factors to which the business is exposed, and accounting or regulatory requirements influence the types of risk measures used by different market participants.
  + Banks use risk tools to assess the extent of any liquidity and asset/liability mismatch, the probability of losses in their investment portfolios, their overall leverage ratio, interest rate sensitivities, and the risk to economic capital
  + Asset managers’ use of risk tools focuses primarily on volatility, probability of loss, or the probability of underperforming a benchmark.
  + Pension funds use risk measures to evaluate asset/liability mismatch and surplus at risk.
  + Property and casualty insurers use sensitivity and exposure measures to ensure exposures remain within defined asset allocation ranges. They use economic capital and VaR measures to estimate the impairment in the event of a catastrophic loss. They use scenario analysis to stress the market risks and insurance risks simultaneously.
  + Life insurers use risk measures to assess the exposures of the investment portfolio and the annuity liability, the extent of any asset/liability mismatch, and the potential stress losses based on the differences between the assets in which they have invested and the liabilities resulting from the insurance contracts they have written.

1. **What are the benefits and shortcomings of each?**

As above

1. **Assuming returns are normally distributed, what are the odds that one-day returns exceed one standard deviation from the mean?**

68% observation within 1sd range

32% outside -> 16% on the upside

1. **Two standard deviations?**

95% observation within 2sd range

5% outside -> 2.5% on the upside

## Interest rate risk

1. **What is interest rate risk?**

Interest rate risk in the banking book (IRRBB) refers to the current or prospective risk to the bank’s capital and earnings arising from adverse movements in interest rates that affect the bank’s banking book positions. When interest rates change, the present value and timing of future cash flows change. This in turn changes the underlying value of a bank’s assets, liabilities and off-balance sheet items and hence its economic value. Changes in interest rates also affect a bank’s earnings by altering interest rate-sensitive income and expenses, affecting its net interest income (NII). Excessive IRRBB can pose a significant threat to a bank’s current capital base and/or future earnings if not managed appropriately. A more detailed description of IRRBB and its management techniques can be found in SRP98.

1. **What is re-investment risk?**

Reinvestment risk refers to the possibility that an investor will be unable to reinvest cash flows received from an investment, such as coupon payments or interest, at a rate comparable to their current rate of return. This new rate is called the reinvestment rate.

Zero-coupon bonds (Z-bonds) are the only type of fixed-income security to have no inherent investment risk since they issue no coupon payments throughout their lives.

**KEY TAKEAWAYS**

* Reinvestment risk is the chance that cash flows received from an investment will earn less when put to use in a new investment.
* Callable bonds are especially vulnerable to reinvestment risk because these bonds are typically redeemed when interest rates decline.
* Methods to mitigate reinvestment risk include the use of non-callable bonds, zero-coupon instruments, long-term securities, bond ladders, and actively managed bond funds.

1. **How do these two interact?**

Fixed income securities such as bonds are instruments that typically pay interest, called the coupon, throughout their lifetimes and then return the face value at maturity. There's generally less risk than with other investments that investors will lose all their money when investing in bonds. There are still risks associated with bonds, though, including factors called interest rate risk and reinvestment risk that focus on whether investors could ultimately be worse off due to changing market conditions.