	Index Information			<u> </u>		lanning Tim		Ex	ecution Time	_		Physical Plan			Behaviour Expla	nation Confid	uration Tuning
Query No. Query	Variable Indexed Type of	of Index Used In	ndex Command	Justification	Before	After	Difference	Before	After	Difference	Before	Slowest Part + Explanation	After	Slowest Part + Explanation	Physical Plan	Timing Commands Us	ed Explanation
union - from load, does not good from the control of the control o	The first index (obs., \$1, \$2pp;, cd., lesse) is considered to the \$4pp;, cd. columns in the columns of the \$1, \$2pp; cd. columns in the columns of the colu	id bt fo (h B-Tree	CREATE INDEX x_fg_fdgrp_desc tree ON	The first index helps the database quickly be a search of the search of	0.795	0.138	0.657	236	0.13	2.230	Data Output Explain Messages Notifications  OUTPUT PLAN  Intel  1 tests Author (1-13.1.26.45 rows-298 width-178) (actual stone-0.455.2.144 rows-493 loops-1)  1 tests Author (1-13.1.26.45 rows-298 width-178) (actual stone-0.455.2.144 rows-493 loops-1)  1 tests Author (1-13.1.26.45 rows-298 width-178) (actual stone-0.016.0.460 rows-7144 loops-1)  3 tests (1-13.1.26.45 rows-1.13.1.26.45 rows-7144 width-151) (actual stone-0.016.0.460 rows-7144 loops-1)  4 * Hath (1-13.1.26 rows-1.13.1.26 rows-1.14 width-2017.0.019 rows-1.100ps-1)  5 Buckets (1-13.1.26 rows-1.13.1.26 rows-1.14 rows-1.14 width-2017.0.019 rows-1.100ps-1)  5 Buckets (1-13.1.26 rows-1.14 ro	The Hash Join is the slowest pair of the execution. It processed to the execution it processed to the execution is processed to the execution continues tool, date and fit group made tool, date and fit group made to the execution of the executio	Data Output Explain Mestages Notifications  OUTP / NA  of the Company of the Comp	The Nested Loop join is the showed part of the execution to the condition of the condition of the condition of the condition of the complete.	Before indexing, the query used a hash join, resying on sequential fit group labels, the property of the prope	a in the same of indices reduced execution set and planing time enable_seqscame significantly.	Indiax was not used so what to turn of the sequential of contraction of the country planner to utilize the index.
2 salect from weight where seq = "T".	The seq is the column indexed in the weight sold.	C Gid	CREATE INDEX bx, weight, seq_b tree ON weight (seq);	This index helps the database quickly search for records based on the value of seq in the weight table.	0.258	0.177	0.081	1.39	1.165	0.225	Data Output Explain Messages Notifications  QUERY PLAN  I text  1 Seq Scan on weight (cost=0.00.287.61 rows=6666 width=50) (actual time=0.033.1.200 rows=6666 loops=1)  Filter. (seq = 1":bpchar)  Rows Removed by Filter 5343  4 Planning Time: 0.258 ms  5 Execution Time: 1.390 ms	The Sequential Scan on the weight table is the storeot part weight table in the storeot part. This occurs because there's no indice on the seq column, so PostgreSCL hat to scan all 6,666 rows to filter for seq = "1".	Outo Output Explain Messages Notifications  OURSP FLAN  A text feet place Size on everytic (cost-75.95.288.27 rows-6666 width-00) (schall time-0.185.0.916 rows-6666 loops-1)  3 freep Source cest-124  4 Billion before Size on everytic (cost-7.95.288.27 rows-6666 width-00) (schall time-0.185.0.916 rows-6666 loops-1)  5 feet Cost Cost Cost Cost Cost Cost Cost Cos	The Bilmap Heap Scan is the strovest part. salaring 0.731 ms as a involvest part salaring 0.731 ms as a involvest factoring using the bilmap operated by the index scan. It requires peaking 122 heap blocks (as indicated in Heap Blocks: associe 124).	Before indexing, the query used a sequential scan on the 'weight' table, scanning at lower and the little, scanning at lower and the little process and the litt	a  The use of the direct reduced set plants are displaced set plants grant set plants grants.	Index was not used so we had to turn off the separation of the separation for the term of the separation to force the quantities the index.
select * from weight where sleet = "I and allowed * 2"," and allowed * 2", and	The 'see' and The 'encount are the columns indexed in the 'weight' table.	C id	CREATE INDEX fx, weight, seq. a nount, brise ON weight (seq. amount);	This composite index optimizes both conditions in the query, reducing science and improving pathormanic relevant rows.	0.135	0.122	0.013	1.113	0.05	1.083	Data Output Explain Messages Notifications  QUERY PLAN    text     Seq Scan on weight (cost-0.00.320.13 rows-144 width-50) (actual time-0.013.1.097 rows-60 loops-1)   Rows Removed by Filter. 12949  4 Planning Time 0.135 ms  5 Execution Time: 1.113 ms	The Sequential Scan on the weight table in the slowest part weight table in the slowest part because the slowest part of the s	Data Output   Explain   Messages   Notifications	The Bilmap Heap Scan is the sixwest part taking 0.021 ms due to the necessity of due to the necessity of affecting with the index.	Bafore indexing the query used sequential scale on the weight as the control of the weight that the fill the fill the sequential scale on the weight will be set to the sequential scale on the weight and the consulting on the sequential sequen	The use of the y index reduced execution and planing time significantly.	Index was used.
select ridb, yo from every ment of the control of t	The nitt, no column is indeed in both the weight and food, this lattice.	id w	CREATE INDEX bx weight, ndb, n o, btree On veight (ndb, no); CREATE INDEX bx, food, diss, nd b, no, btree On food diss, nd (ndb_no);	The first index helps the distablese quickly search for records of right, not in the weight table, optimizing the join between nut_data and weight. The second index helps the distance in the	0.063	0.135	-0.072	7.31	6.73	0.580	Data Output Explain Messages Notifications  ☐ UERY PLAN ☐ 191 ☐ 1	The Sequential Scan on the weight table is the slowest operation taking older and us to peration taking older and us to scanning 13,000 rows.	One Output Explain   Message   Motifications	The Append operation is the slowest operation teleng slowest operation teleng process the cultipart of both process the cultipart of both subsparies, resulting in 20,155 must needing to be consuming in 20,155 must needing to be consuming.	Before indexing, the query plan used Studyaey Scans with Sequential Scans on the weight Sequential Scans on the weight fall scans of 13,000 and 7,140 fall scans of 13,000 and 13	increase as the end database set evaluates their enable_seqscan=	One of the indices was not clusted to we had to sum of the stack to sum of the stack to utilize the index.
select rule, yo from swight except select cless, ye from delevery.	The nds, no column is indeed in both the weight and distinct tolone.	B-Tree	CREATE INDEX Xx, weight, ridb, ri 0, between ON (ndb_no); CREATE INDEX dx_dateridn_ridb no, between On atteridn (ndb_no);	These indices optimize quaries that filler or compare based on the not, so colors, and possible of the colors of t	0.165	0.009	0.096	33.688	30.617	3.071	Data Output   Explain   Messages   Notifications	The Append operation is the stowest operation laking 19.195 ms as it needs to process the operated of the history of the operated of the history in the operation of the operati	Data Output   Epilon   Messages   Notifications	The Append operation is the slowest operation toking 16.175 mile at the residence of the subsparence, resulting in 108,854 more needing to 108,854 mor	Before indexing, the query used sequential scale on the "weight" 15.000 and 03.845 inext, results were expected. The control were expected, the control were operation, adding computational overhead and sideding to a painting firm of 0.165 ms and other control of the control o	The way of priction and the standed execution and and planing time and planing time and planing time.    The way of priction and planing time and planing time and planing time.	Indian was not used to we had to ten or had to ten of the sequential stress to force the query planner to written the indian.
subset maked numbers of the product	The lagrams and rist, no columns are obtained in the risk, get fallow.  The risk year and risk, you columns are individed in the risk, and risk year and risk year.  The risk year columns is interested in the weight above.	B-Trees C	CREATE INDEX  L. ruit of lappe  and the complete	The first index helps the distribute quickly find to control quickly distribute quickly find to control quickly and the distribute quickly find to control quickly find to	1.004	0.785	0.219	13.839	4.311	9.528	Dubbs Output   Explain   Messages   Motifications	The Hash Join is the scheed part of the physical plan is these and the physical plan is the skin plan physical physical plan physical plan physical ph	Color Color Esplain Messages Netifications  GENTAL  JOHN TAIL  JOH	The Habit Joint numbers the street of the Street part over after the heat of the Street part over after the heat of the Street part over after part of the Street part of the Street part over the Str	Before lockering the planner was required by schoring not, and an analysis of the schoring not, and and plan however after turning of which have been a size turning of which have been a size of the schoring of the schoring of the schoring of the the schoring of the schoring of the the schoring of the schoring of the the schoring of the schoring of the the schoring of the schoring of the the schoring of the schoring of the the schoring of the schoring of the the schoring of the schoring of the schoring of the schoring	The use of indices at a constant of the cons	Index sea Art Josef to see that to have of the sequential of a sequential season of the season of the season of the season of the VIII.se the index.

salect mather national mathematic	The laguene and net as so odurine are included in the size of the	B-Tree	CREATE INDEX ISS., unit, of I Japanis, and	'nut_data', as well as	0.474	0.730	-0.256	45.871	14.582 3	31.089	Data Output   Explain   Messages   Notifications	The Nach John is the streeted part of the Streeted	Best Organ   Ciplan   Messages   Norfoctions	The Heat, Just constant for all sources of the second of the end of the second of the	Online including the query visits may be presented a construction of the present of the construction of displayers and di	tme, but planning time may slightly	incide was not disable to a high property of the control of the co
salact count(*) from tool, day where the large to the salace the large to the large through the large	the lat_factor column is indexed in the food_dea table.	B-Tree	CREATE INDEX Idx, food, das, fat factor, ruli factor, ruli factor, ruli food, das (lat, factor) WHERE fat, factor IS NULL;	We used the index to optimize the quely through creating a specifically for one specifically for more specifically for more where fall, factor 15 MULL ; PostgreySDL can more efficiently locate and count the matching rows without the count of the matching rows without the count to be read to be sourced to query performance by reducing the number of rows that meet to be sourced and by is everaging the specific performance of the second can be sourced to be sourced and by is everaging the specific performance of the second can be seen to be seen t	0.068	0.109	-0.041	0.949	0.249	0.700	Data Output   Explain   Messages   Notifications	The Sequential Scan processes: 1911 roses from the large continues of the large continues o	Data Output Explain Messages Notifications GUIRFMAN At ret 1 Aggregate growth St. (26.17 new 1 willing 1) (solution of 222.5.237 new 1 larger) 2 - Annother Guirmann (solution of 222.5.237 new 1 larger) 3 - Notice (1) Solution growth (solution of 222.5.237 new 1 larger) 3 - Notice (1) Solution (solution of 222.5.237 new 1 larger) 5 - Description (2.247 new 1) 6 - Description (2.247 new 1) 7 - D	The Index Scan operation is all the physical pins as it scene: 1311 rose from flood, disc scene; 1311 rose from flood, disc scene; 1311 rose from flood, disc scene; 1311 rose from flood, disc scan flood, disc s	Before inspecting a sequential or an extension of food, less all distances from the food of the self of the food of the self of the food of the self of the food o	The use of indices as significantly enduce execution first, but planning first may significantly enduce execution first, but planning the manufacture of the evaluation their usefulness. This consignification is considered to the execution time gains.	Index was not used of the sequential can to force the quey planner to units the roter.
select count(*) from food, does where the feet or in mult.	The fail factor column is indexed in the Good, dies table:	8-Tree	CREATE INDEX Idv, food, des, for factor null bites ON food, des (lat, factor) WHERE sta, factor IS NULL;	This partial index optimizes the quary brond primities the quary brondy including rows where fat, factor IS NULL, thus making the quary more efficient. Instead of scanning the entire table, PostgreSQL can use this index to quickly find and count only the rows where the index to quickly find and count only the rows where the index to quickly find and count only the rows where the index to quickly find and count only the rows where the index to quickly find and count only the rows where the index of the rows where the properties of the rows where the rows w	0.072	0.125	-0.053	0.908	0.465	0.443	Data   Output   Explain   Messages   Notifications	The Sequential Scan is the stowed operation as it scars showed operation as it scars showed operation as it scars the with filtering condition fat, Sactor is null which consumed 0.829 ms.	Data Output Epilan Messages Notifications  OUTPY-NA  July 10  Throughing bear or hout, also (and-\$231.202 Gross-1111 width-151) (abut fire-\$133.632 coon-1911 tops-1)  \$ Reston Coor (Michael MALL)  A regulated assert of A	The Bitmap Heap Scan is an sixteest part of the physical plan as a firm-hore scanning 1911 rose of the foot part of the part of the foot part of the fo	Before indexing, a sequential so was used on the Tood, east table to filter rows where I fat, feed or I was I was a sea of the Tood, east table to filter rows where I fat, feed or I was	an The use of the soft registrating restriction of the soft registrating restriction of the soft registration of the soft	Index was used.
salect min(r), factor) from flood, data willeam fat, factor in mult,	The fast Jacker and a factor columns are indexed in the food, dee table.	B-Trise	CREATE INDEX ids, food, das far factor, in a factor, on a food, das (lat. factor, mactor) WHERE tat, factor is NULL;	Trisi partial index includes only rows where fat, factor 15 NULL and crovers both the filtering condition of the Minute for the Min	n n n n n n n n n n n n n n n n n n n	0.231	-0.119	1.003	0.292	0.711	Data Output   Explain   Messages   Notifications	The Sequential Scan is the slowest operation as it scars 15th town of the bod, due table in any 1st and 1st an	Data Cuptat Egilan Messages Notifications OMER TANA 1 Appropria (mork 512 Ak 37 more) - settled planet stree-22k 1.532 more) Image-1 2 = hote of histomerapia, benefits, fire, fire, red. a, from 28k of settled planet stree-22k 1.532 more) Image-1 3 = hote of histomerapia, benefits, fire, fire, red. a, from 28k of settled planet stree-23k 5.517 more 4 = Patrony Time 5.219 m. 5   Excellent Time 4.202 m.	The Index Scian operation is all the allowed operation in the proposal part in the proposal part in the proposal part in the Castle and Castle	Defore indexing a sequential so: was performed on food, des table sequential so to table sequential so table sequential s	The use of the index significantly index significantly for the property of the	Index was used.
select sumin factor) 11 Son lood, as where led, flector in mail,	The fat, factor and n factor columns are inchessed in the food, dee stake.	B-Tree	CREATE INDEX idx, food, das, fat factor, mil. of clor, bree ON factor, bree ON fat, factor is NULL;	This partial index includes only the room where fat, factor IS NULL, and it also includes the nichodes that nichodes the nichodes the nichodes that nichodes that nichodes the nichodes that nichodes the nichodes that n	0.053	0.121	-0.088	0.883	0.274	0.609	Data Output	The Sequential Scan is the slowest operation as it scan are some statement of scan are some some statement of the scan are some some some some some some some som	Data Output Englain Messages Notifications OCRETIFIAN 1 Appropria (mice 437, 6, 47 mose) - seath-of-prinal time of 327, 638 mose) loope-1) 2 = hate of historiana grap, hot, fine, fine, fine, fine, prince and lood, des (mice 22, 66.54 mose) 1111 with-of-(install time of 015, 0.31 mose) 4   Haurring Prince 1317 ms 5   Excellent Test 4224 ms	The Index Scan operation is at the scient operation with the scient operation in the 1911 route from long data where fall factor is not for the 1911 route fall factor is not consuming operation in all the physical plan taking 0.15 ms.	Before indexing a sequential social was performed on food, dies tate for the performed on food, dies tate for the performed before in deal the file section in the performed which is a bottomer performed which is a bottomer performed with a better the performed with a better to the performed on the result of the performed with a better to the performed on the result of the performed with	The use of the index aggrécanty execution states aggrécanty reduce execution firme, but planning firme may aligney in increase as the evaluation their unefulness. This minor ownlead is in negligible to the case of the case	Index was used.
salect summit, factor) 12 then food, deal where full_dealer is not mail.	The fat, factor and n, factor columns are indexed in the fixed_cas table.	B-Tree	CREATE INDEX idx, food, das, fat factor, fort, factor, fort, n, factor, bitse ON food, das (lat, factor, fat, factor IS NOT NULL;	This partial index includes only the flow where fat, factor 15 MOT NULL and cover both the filtering received in the filtering received in the filtering received in the SUM function (n. factor). This allows Thousand the rows that meet the condition, bypassing the reset to be condition, bypassing the reset the condition, bypassing the reset to be condition, bypassing the reset to be conditionated by the reset	5 S S S S S S S S S S S S S S S S S S S	0.296	-0.194	0.983	0.717	0.248	Data Output Explain Messages Notifications	The Sequential Scan is the abovest operation as it scan abovest operation as it scan above the scan	Dati Odgal Espilan Massayan Nadiriaturus	The Index Scian operation is  B his slowed operation in Scian Scientific Scian Scian Scian Scian Scian Scian Scian Scian Scian Scientific S	Before intensity a sequential of the service of the	The use of the sin landar significantly and single significantly and single significantly and single significant s	Index was used.
select tagname from pan until de man pan until de mande de man pan until de man pan until de man	dative displacement, july	biree	oreania Index las, dist Jedancia Jones on delarios (datasero, id);	dataser; jel is almeady primary key in data, ze primary key in data, ze datasetchi (dal) so we added ain inder the hopes of some primary and pin instead of comps in seasod ones nessed loops	a rc rc c c c c c c c c c c c c c c c c	0.366	0.338	2.821	0.212	2.609	COURT FLAM  The Montal Case (Section 2.7. 2446.31 move 256 width (Securial serve 2212. 2.74 move 27 loops 1)  - Securial Case (Section 2.7. 2446.31 move 256 width (Securial serve 2212. 2.74 move 27 loops 1)  - Securial Case (Section 2.7. 2446.31 move 256 width (Section 2.76 move 27 loops 1)  - Section 2.75 width (Section 2.7	The abovest pair of this Cucay is distanced, piley rating ready 2.5 weeks (continued to the continued to the continued to the continued to access to the continued to access on the rest to access on the continued to access the continued to access to access the continued to access	00977 M.  ***CALL TO A T	There isn't a specific part the balance on situating long time. In the balance on situating long time that the law poor of previous of the law poor of previous of the law poor of the law poo	After bossing the planner and to be and appeared and appeared and appeared across were companied and appeared across were companied and the state. As the state of the state o	56 The change from Plantage removes and plantage re	indus was not studied as we had to bee of the sequential of the se

salect * from weight salect mounts 2;	weight(amount)	biín	create index idx, weight, amount, nt, brin on weight using BRIN (amount);	the fatering condition of SRI states as their of SRI states and we say the say the say of the say o	n N N d d d o 0.052	0.386	-0314	1.123	0.212 0.91	0.000 F/MA	There's only one operation in this gray which is a sequential soon service of the	DOES FLANT  White the contribution for the contribution of the con	There ain't a specific part the tong which suggeste that the is a good physical plane for the guery specifically showever guery specifically showever to be account for the use of the specific part of the specific part of the specific part of the specific part of problems be not temporal orines.	After forming the planner not to a sequential scare, the nested for an experiment scare, the nested for an experiment scare were series of the scare was a series of the scare from the sc	The change from floatine well-de- formed to the change of	Index was not used so we had to turn of the sequential for sequential for the sequential
15 added * Born weight where arround > 2;	weight(amount)	brin	create index idx, weight, amount, bin on weight using BRIN (amount);	the filtering condition made us think of BRH indices and we made the pro- blement sile prices and pro- planner sile prices of the pro- cess of the prices of the prices of the was not a between range quary but a greater than one.	n N d d of 0.049	0.057	-0.01	0.845	0.984 -0.12	CACCO FAMA  Sept East an example panel do 22 ft it sees 1715 with HV-00 (actual tree 1811 180 (recer 1715 loops 1)  Find in proceeding Find 17154  Finds a proceeding Finds 17154  Finds a proceedi	There's only one operation in this quary which is a sequential scen to fifter the elements. The makes amplifying of the filtering condition.	ORDET FLAN The spin of from a couple () and 13.4 200 15 coup 2713 with 401 gas of Beautine 14.50 men 1711 taper () Table to the greaters - T shall present (a for a formal from Annie Anni	Here the Bitmap heap scan on weight in the bringet parties the from the execution. This takes from the execution. This would make serial because most of the data is being vernoved by the filter.	Initially the planner still worst for the planner still pl	The disapped the trend of an exist in significant changes and an exist in significant changes are continued as execution. This makes series at the sursteed of yours being filtered and the sursteed of yours being filtered and the sursteed of the sursteed	Index was not used so we had to turn of the sequential query planner to utilize the index.
select mare, cleac, avg (amount) from weight where amount > 2 group by amounts, mire, cleac.	weight(amount)	brin	create index idx, weight, amount, nt, bin on weight using BRIM (amount);	the littering condition made us think of BRU indices and we made the they would help the planner skip churks date especially as it was not a between range query but a less than one.	n N d d of 0.147	0.057	0.090	1.688	1.176 0.512	OCCUPANI  WANDAY WAS DESCRIPTED AS 30 Security 20 wilds 20 (instal street 400.1 SM cover140 logors)  Wanday was present start, date  State for a present start, date  State for a present start, date  Facilities a State or 1900 and 1900 an	The sequential scan on weight is The sequential scan on weight is acaucitor from else which makes sense because its removing mos of the rows.	October Acad	Here the Bitmap heap scan on weight is the longest part of the data heavy most of the data is being sense to be a long sense to be the longest part of the data is being sense to be the little.	trisially the planner still went for sequential scan so after forcer; is sequential scan so after forcer; is timely taken some wife or created index to filter and a timel some scan on veget fails for settlements.	The changes this free did not result in a significant in a significant did not result in a sig	Index was not used to what to ham of the seguential scale to force the query planned to utilize the index.
select fil large, deac, not seem food, and it comes from the file of the file	nd_dddphom_thdes)	hash	create index	By creating a head index on numbers statistics on the state of the sta	6 0.331	0.236	0.025	73.37	28.038 45.33	Description   Comparison   Co	The foregoed part of this casery is the parallel scaparatide control on the parallel scaparatide control on the case of the service of the service process on the service process on the service process on the service process of th	Column   C	Have the languary part is once again a parallel sequential sequential against the second of the against the second of the against the second of parallel second of the second of the second of languary to languary to l	After transducing the hash holes they place in standard preferring applications may apply and applications may apply and applications may be apply and apply a	The introduction of the interest and a secondary from concept the interest and a secondary from concept from	India was used
select *from fd, group fd  18 on fd fdgap_cd = fod.  Sprp_cd = fod.	food_des(fdgrp_crl)	hash	create index idx, food, des_food one, food, des using hash (fdgrp_cd);	This index was madely to decrease leitminate the hashing on the fly time and that yet speeding the query used to the time and the time and the time.	0.210	0.210	0.000	2.963	2.907 0.050	DUSTY PLAN	The sequential scan on food, dis- tribution of the sequential scan on food, which is probably because its going through 7146 rows.	OUDN PLAN  Institution (control \$1.00 dd control \$4.4 within 170) (setual times 0.04.2.489 control \$4.00 pt 1)  Institution (control \$1.00 dd control \$4.4 within 170) (setual times 0.04.2.489 control \$4.00 pt 1)  Institution (control \$4.00 pt 1	The longest operation in this curve year year on the nested loop and this probably because operations are operations are operations are operations are operations of years of the probability of the control of the cont	Because the hashing was alread handled by our index, the planne option for a nested loop with billion the oversiching structure is the same	y Time was properly the part of the part o	Index was not used so we had to turn of the sequential to query planner to utilize the index.
salesc * from fit group is over pin flood, due fod over pin flood, due fod 19 kgy, cd * floot over pin nut, dash at on over pin nut, dash at on over pin nut, dash at on other pin nut, dash at on other pin nut, dash at on	tood_seeptigypy_cd)	hash	create index ide, food, des, fog rp, cd, hash cn food, des uning hash(fdgrp_cd);	This index was made to decrease elements the hashing on the fly time and that years are specified to the specified of the specified the query uses the specified of the specified of the property of the specified	0.242 p	0.250	-0.008	141.295	157.406 -16.11	GART PLAN	The half join is by fur the longest single operation in the conditions of the condition of the condition operations. This is because of the where number of the condition operations in the condition operation. The individual operation operation operations are sufficiently as the condition of the condition operation of the condition operation operation operations of the condition of the condition operation of the condition operation of the condition operation of the condition of	OCENTRAN  OCENTR	structure of the query wasn' changed, sequential scans were just substitued for inde	Girgely, the sequential source became index source in the attributes we are desting with sequentiary layer as choosed make primary layer as devoted make index of source index index of the attributes we are desting with the control of the contr	Transplants  Trans	No matter what we did the index works of the useful of the useful at a for the case of the useful at a forth of

											QUERY PLAN		QUERY PLAN					
											HashSetCp Intersect (cost+0.00, 549.69 rows+1 width+28) (actual time+1.458.1.461 rows+0 loops+1)		HashSetOp Intersect (costr0.00.475.91 rowsr1 widthr28) (actual timer0.702.0.703 rowsr0 loopsr1)					
											> Appand (cost=0.00, 548.94 rows=299 width=28) (actual time=1.256.1.454 rows=118 (acpa=1)		-> Append (cost*0.03, 475, 16 tows*299 width*281 (actual time*0.659, 0.696 rows*118 (cops*1)					
													<ul> <li>Subgrey Scan on "SELECT" 1" (cost=0.00, 287.62 nows=1 width=16) (actual time=0.622, 0.622 nows=0 loops=1)</li> </ul>					1
1				1	1			1			-> Subquery Scan on "*SELECT" 1" (cost*0.00.287.62 rows*1 width*10) (actual time*0.576.0.576 rows*0 loops*1)		Seq Scan on weight (cost=0.00.287.61 rows=1 width=6) (actual time=0.621.0.621 rows=0 (oops=1))		1		1	
											→ Seq Scan on weight (cost=0.00.287.61 rows=1 width=6) (actual time=0.575.0.575 rows=0 (oops=1)		Filter (arreant > 30' double precision)					
	(select ndb_no from										Fifte: (amount > '50':double precision)		Rows Removed by Filter, 13009					
	weight where amount >										Rows Removed by Filter: 12009		- Subquery Scan on "SELECT" 2" (cost=10.21186.04 rows=298 width=10) (actual time=0.036.0.068 rows=118 loops=1)			Timing		
	INTERSECT			create index idx_food_des_fdg	This index was made						→ Subquery Scan on "*SELECT* 2" (cost=1.31.259.83 rows=298 width=10) (actual time=0.679.0.871 rows=118 loops=1)	The longest single elements of	→ Needed Loop   Cont+10.31.182.06 rows+298 width+6) (actual time+0.035.0.061 rows+118 loops+1)	The longest single operation in this query plan is bitmap	The only change that happened	significantly changed by the		
20	(select ndb_no from food_des fod inner join	food_des(fdgrp_cd)	hash	rp od hash on	the hashing on the fly	0.139	0.186	-0.047	1.485	0.730	55 (Hash Join (post=1,31,256.85 rows=298 width=6) (actual time=5.678.0.962 rows=118 (cops=1)	this query are hash join and			was that the sequential scan on food_des became a bitmap index	use of bitmap	none	Index was used
	fd_group fd on fd. fdgrp_cd = fod.fdgrp_cd			food_des using	time and that way						Hash Cond (Tod fidgs, cd = fd fidgs, cd)	sequential scans on food_des	→ Beg Scan on M, group M (cost+0.00.1.30 rows+1 width+5) (actual time+0.014.0.014 rows+1 loops+1) Filter: (fiding, desc + Snacks/:test)	was probably because of it going through 118 rows.	scan			
	fdgrp_cd = fod.fdgrp_cd			hash(fdgrp_cd);	speeding the query up						→ Seq Scan on food_des fod   cost*0.00.233.46 rows*7146 width*11) (actual sme*0.010.0.363 rows*7146 loops*1)		Bons Removed by Filter, 23	going shough 116 rows.		instead of sequential scans		
	where fd.fddrp_desc = 'Snacks'):										> Hissih (cost=1.50.1.30 rows=1 width=3) (actual time=0.009.0.009 rows=1 loops=1)					1 '		
											Buckets: 1004 Batches: 1 Memory Usage: 9kB							
											-> Seq Scan on fd_group fd (cost=0.00.1.30 reves=1 width=5) (actual time=0.005.0.006 reves=1 loops=1)		Recheck Cond: (Ydgrp_cd = fd.fdgrp_cd)					
											Filter (Idday, desc = Snacks: test)		Heap Blocks: enact=10					
													→ Bitmap Index Scan on idx, food, des, fdgrp,cd, hash (cost=0.00.10.23 rows=296 width=0) (actual time=0.009.0.009 rows=118 loops=1)					
											Rows Removed by Filter: 23		Index Cond. (Mgyp.od = M.Mgyp.od)					
											Planning Time: 0.199 ms		Planning Time: 0.186 ms					
											Execution Time: 1.465 ms		Execution Time: 0.730 ms					
											QUERY PLAN		QUERY PLAN					
											text		text	•				
													Aggregate (cost=937.94.937.95 rows=1 width=32) (actual time=1.872.1.873 rows=1 loops=1)					
l				1	1			1			Aggregate (cost=274.94274.95 rows=1 width=32) (actual time=2.2112.213 rows=1 loops=1)		→ Nested Loop (cost=7.61.920.08 rows=7146 width=6) (actual time=0.0211.376 rows=7146 loops=1)		1	1	1	
l											-> Hash Join (cost=1.54.257.07 rows=7146 width=6) (actual time=0.017.1.403 rows=7146 loops=1)				Because the hashing was already	1	1	
	1			1	1			1					→ Index Only Scan using fd_group_pkey on fd_group fd (cost=0.1412.50 rows=24 width=5) (actual time=0.003.0.009 rows=24 loops=1)			1	1	
	select maxindb no) from			create index	This index was made			1			Hash Cond: (fod.fdgrp_cd = fd.fdgrp_cd)		Heap Fetches: 24	The longest operation in this		Time was slightly	1	Index was not us so we had to tur
l	(select * from fd_group fd inner join food_des			idx food des fdo	to decrease/eliminate						> Seg Scan on food des fod (cost=0.00233.46 rows=7146 width=11) (actual time=0.0040.399 rows=7146 loops=1)	The longest single elements of	>> Sitmap Heap Scan on food, des fod (cost+7.4834.84 rows+298 width+11) (actual time+0.0090.041 rows+298 loops+24)	query is now the nested loop and thats ninhably herause	handled by our index, the planner onted for a nested loop with hitms	affected by the sp swapping of	set	so we had to tur off the sequentia
21	for on fri from ord = fort	food_des(fdgrp_cd)	hash	rp_cd_hash on food_des using	the hashing on the fly time and that way	0.118	0.135	-0.02	2.233	1.898		this query are hash join and sequential scans on food_des		it's internal children	opted for a nested loop with bitms heap and index scars. However	iterative and	enable_seqscan=of	ff: scan to force the
l	fod on fd.fdgrp_cd = fod. fdgrp_cd) as t;			hash(fdgrp_cd);	speeding the query up			/			Hash (cost=1.241.24 rows=24 width=5) (actual time=0.0080.008 rows=24 loops=1)	anquerium acuma cir 1000_005	Recheck Cond: (fdgrp_cd = fd.fdgrp_cd)	operations are operating on less tuples	the overarching structure is the same	hashing algorithms		query planner to utilize the index
l				1				1			Buckets: 1024 Batches: 1 Memory Usage: 9kB		Heap Blocks: exact=244				1	Daniel and Aldex
				1									→ Bitmap Index Scan on idx_food_des_fdgrp_cd_hash (cost+0.00.7.40 rows+298 width+0) (actual time+0.007.0.007 rows+298 loops+					
								, 1			Seg Scan on fd. group fd (cost=0.001.24 rows=24 width=5) (actual time=0.0020.004 rows=24 loops=1)	loops=1)	Index Cond: (fdgrp_cd = fd.fdgrp_cd)					
					1						Planning Time: 0.118 ms							
											Flating fille, 0.116 lbs		Planning Time: 0.135 ms					
											Execution Time: 2.233 ms		Execution Time: 1.898 ms					
													Ti control of the con					
											QUERY PLAN	<u> </u>	QUERY PLAN	-			1	
					1			4 '			text		QUENT PLAN	4				
											HashAggregate (cost=310.67382.13 rows=7146 width=38) (actual time=3.4934.397 rows=7146 loops=1)		HashAggregate (cost=448.94.520.40 rows=7146 width=38) (actual time=4.2705.172 rows=7146 loops=1)					
											Group Key: fod.ndb.no							
													Group Key: fod.ndb_no					
											Batches: 1 Memory Usage: 1425kB		Batches: 1 Memory Usage: 1425kB		Simply, the sequential scans			No matter what
	select ndb_no, sum (fat_factor), sum										→ Hash Join (cost=1.54257.07 rows=7146 width=22) (actual time=0.0211.801 rows=7146 loops=1)			The slowest part again was	became Index scens on the	Timing wasn't		did the index wo
				create index idy food des file	This index was made to decrease/eliminate							The longest single elements of	<ul> <li>→ Hash Join   cost=13.08.395.34 rows=7146 width=22) (actual time=0.020.2.405 rows=7146 loops=1)</li> </ul>	the hash join and this makes sense because the actual		significantly	set enable_seqscan=of set	not be used. This ff: supposts that the
22	from fd_group fd inner join food_des fod on fd.	food_des(fdgrp_cd)	hash	rp_cd_hash on	the hashing on the fly	0.281	0.146	0.135	4.732	5.468	Hash Cond: (fod.fdgrp_cd = fd.fdgrp_cd)	this query are hash join and	Hash Cond: (fod.fdgrp_cd = fd.fdgrp_cd)	structure of the query wasn't	Postgres. This is because most of the attributes we are dealing with are primary keys so it would make	changed by the	set	
	fdgrp_cd = fod.fdgrp_cd)			food_des using hash(fdgrp_cd);	time and that way speeding the query up						→ Seq Scan on food_des fod (cost=0.00233.46 rows=7146 width=27) (actual time=0.0050.973 rows=7146 loops=1)	sequential scans on food_des	→ Index Scan using food_des_pkey on food_des fod (cost=0.28.360.47 rows=7146 width=27) (actual time=0.007.0.916 rows=7146 loops.	changed, sequential scans	are primary keys so it would make	scans instead of	enable_indexonlyso	can useful at all to th
	as t group by fat_factor,			main(ragrp_ca),	specially one query up								-> Hash (cost=12.50.12.50 rows=24 width=5) (actual time=0.008.0.009 rows=24 loops=1)	scans	sense for the planner to choose them over a user created index or	n sequential scans	- 011,	query and the planner preffere
	pro_factor, ndb_no;										<ul> <li>Hash (cost=1.24.1.24 rows=24 width=5) (actual time=0.0090.010 rows=24 loops=1)</li> </ul>				another attribute.			other indices.
											Buckets: 1024 Batches: 1 Memory Usage: 9kB		Buckets: 1024 Batches: 1 Memory Usage: 9kB					
													→ Index Scan using fd_group_pkey on fd_group fd (cost=0.14.12.50 rows=24 width=5) (actual time=0.002.0.005 rows=24 loops=1)					
											<ul> <li>Seq Scan on fd_group fd (cost=0.001.24 rows=24 width=5) (actual time=0.0030.005 rows=24 loops=1)</li> </ul>		Planning Time: 0.146 ms					
											Planning Time: 0.281 ms							
											Execution Time: 4.732 ms		Execution Time: 5.468 ms	J.				
											Execution Time: 4,732 ms	1						
											QUERY PLAN	<b>A</b>	QUERY PLAN	2				
l											Text	_	text	•	1	1	1	
	1			1	1			1			Aggregate (cost=489.32489.33 rows=1 width=16) (actual time=7.6967.701 rows=1 loops=1)		Aggregate (cost=627.59627.60 rows=1 width=16) (actual time=5.3815.384 rows=1 loops=1)		1	1	1	
	1			1	1			1			→ HashAggregate (cost=310.67382.13 rows=7146 width=38) (actual time=5.2377.290 rows=7146 loops=1)		→ HashAggregate (cost=448.94.520.40 rows=7146 width=38) (actual time=3.7375.041 rows=7146 loops=1)		1	1	1	
	select max			create index											1	1	1	
	(total_fat_factor), max (total_pro_factor) from			idx_food_des_fdg rp_od_hash on	The first index was						Group Key, fod.ndb_no		Group Key: fod.ndb_no		1	1	1	
	(soler) non sur			rp_cd_hash on food des usino	done to make the			1			Batches: 1 Memory Usage: 1425kB		Batches: 1 Memory Usage: 1425kB		Simply, the sequential scans	1	1	No matter what
	(select ndb_no, sum (fat_factor) as	I		hash(fdgrp_cd);	innerjoin in the subquery easier by			1					→ Hash Join (cost=13.08.395.34 rows=7146 width=22) (actual time=0.023.2.374 rows=7146 loops=1)	The innest merations analy	became index scans on the indices already created by	Timing slightly	1	did the index wo not be used. Thi
	total_fat_factor, sum (pro_factor) as	1. food desifidano od)		create index	allowing for a faster			1			<ul> <li>Hash Join (cost=1.54.257.07 rows=7146 width=22) (actual time=0.024.2.768 rows=7146 loops=1)</li> </ul>	The longest single elements of this query are hash join and		are the index scans and has	Postgres. This is because most of the attributes we are dealing with	changed by the	set enable seascan	_ suggests that the
23		1. tood_des(tdgrp_cd) 2. food_des(fat_factor.pro_factor.ndb_no)	Btree	late front doe for	hash join instead of	0.303	0.188	0.115	7.869	5.438	Hash Cond: (fod.fdgrp_cd = fd.fdgrp_cd)	this query are hash join and	Hash Cond: (fod.fdgrp_od = fd.fdgrp_od)	joins and this makes sense	the attributes we are dealing with	use of index	off:	index was not useful at all to th
	(select * from fd_group fd inner join food_des			factor_pro_facto	hashing on the fly, while the second index			1				sequential scans on food_des	→ Index Scan using food_des_pkey on food_des fod (cost=0.28.360.47 rows=7146 width=27) (actual time=0.007.0.890 rows=7146 loc	<ul> <li>because the actual structure</li> <li>did not change</li> </ul>	are primary keys so it would make sense for the planner to choose them over a user created index or	sequential scans	1.	guery and the
	fd inner join food_des fod on fd.fdarp_cd = fod.			factor pro facto r ndb no cd btr ee on food des	was made with the			1			<ul> <li>Seq Scan on food_des fod (cost=0.00233.46 rows=7146 width=27) (actual time=0.0080.541 rows=7146 loops=1)</li> </ul>		-> Hash (cost=12.5012.50 rows=24 width=5) (actual time=0.0110.012 rows=24 loops=1)		them over a user created index or	n	1	query and the planner preffere
	fdgrp_cd) as t group by			(fat_factor,	intention of speeding up the group by			1			> Hash (cost=1.241.24 rows=24 width=5) (actual time=0.0100.012 rows=24 loops=1)				another attribute.	1	1	other indices.
	fat_factor, pro_factor, ndb_no) as x;			pro_factor.	ap and group by			1					Buckets: 1024 Batches: 1 Memory Usage: 9k8		1	1	1	
	100 MS X;			nao_noj.	1			1			Buckets: 1024 Batches: 1 Memory Usage: 9kB		Index Only Scan using fd_group_pkey on fd_group fd (cost=0.1412.50 rows=24 width=5) (actual time=0.002.0.006 rows=24 loop		1	1	1	
											<ul> <li>Seq Scan on fd_group fd (cost=0.00.1.24 rows=24 width=5) (actual time=0.0030.005 rows=24 loops=1)</li> </ul>		Heap Fetches: 24		1	1	1	
		I		1	1			1					Planning Time: 0.188 ms		1	1	1	
				1	1			1			Planning Time: 0.303 ms				1	1	1	
				1							Execution Time: 7.869 ms		Execution Time: 5.438 ms			1	1	
				+								_		+		+	+	
				1	1			1			QUERY PLAN			1	1	1	1	
				1	This will select all			1			▲ text	There's only one operation in th	a l	1	1	1	1	There's no use
	select srcod_desc from	None	None	None	values of that column adding an index will	0.043	0.043	0.00	0.024	0.024	1 Seq Scan on src_cd (cost=0.001.10 rows=10 width=44) (actual time=0.0100.012 rows=10 loops=1)	query which is a sequential sca	n,	1	This will select all values of that column adding an index will only	1	nothing	
24	select srccd_desc from	24 src_od, None		None	only increase	0.045	0.040	0.00	0.024	0.024	2 Planning Time: 0.043 ms	which makes sense because we're just projecting		1	increase complexity		- Louising	something that is a sequential sca
24	select srccd_desc from src_cd;											we're just projecting						
24	select srccd_desc from src_cd;				complexity						3 Execution Time: 0.024 ms	we to just projecting			notate company			a sequenta sca
24	select srccd_desc from src_cd;				complexity						-	we to just projecting			notate company			in annual