Designing, Testing & Tuning Objectstore Queries

This document describes best practice for designing queries within FileNet Content Manager

Version: 1.1
Dominik Baer dominik.baer@ch.ibm.com

Contents

1 Document information

1.1 Open points

No table of contents entries found.

1.2 Decisions taken

No table of contents entries found.

1.3 Versions

Version	Changes / Status	Author(s)	Date
1.0	Draft	D.Baer	2.7.2018
1.1	Updated and reformatted	D.Baer	10.3.2021

2 Scope of this document

2.1 Querying meta data

IBM FileNet Content Manager / Case Manager supports searching on metadata either by using:

- content based queries (Fulltext)
- relational queries (DBMS)

2.1.1 Fulltext searches– out of scope

This document doesn't discuss the approach using full text searches, even it might be a good alternative addressing the burden to build numerous database indices to address various search strategies.
For information about building full text based searches see:
CBR queries
Reasons to use content-based queries instead of relational queries are:

- case insensitivity
- highly customized searches with constantly changing search arguments

Reasons not going this approach might include the additional need of maintaining CSS Servers and the need to maintain additional storage for the Full text index.

2.1.2 DBMS searches – in scope

2.1.2.1 Database vendors

An Objectstore is always created within a supported database system like

- MS SQL Server,
- Oracle
- DB2
- PostresSQL

Searches are either been formulated against the content subsystem or the workflow system.

You find further information regarding the preparation steps on the database side
https://www.ibm.com/support/knowledgecenter/en/SSNW2F_5.5.0/com.ibm.p8.planprepare.doc/p8ppi084.htm

Please check the compatibility matrix at
https://www.ibm.com/software/reports/compatibility/clarity/softwareReqsForProduct.html

Most of the discussion in this document is aimed at customers using Oracle. Most techniques can be applied as well on the other database vendors.

2.2 DBMS

2.2.1 Case sensitivity

The case sensitivity depends on the database vendor and version. You find most relevant information googling for “Collation” and your database system vendor.

MS SQL: https://docs.microsoft.com/en-us/sql/relational-databases/collations/collation-and-unicode-support?view=sql-server-ver15

Oracle: https://docs.oracle.com/database/121/NLSPG/ch5lingsort.htm#NLSPG005

DB2: https://www.ibm.com/support/knowledgecenter/SSEPGG_11.5.0/com.ibm.db2.luw.admin.nls.doc/doc/r0050489.html

PostgresSQL: https://www.postgresql.org/docs/13/collation.html

In general MS SQL come out of the box with a case insensitive collation and is not as problematic as DB2 and Oracle. DB2 has introduced case insensitive collations a few years ago and finally Oracle comes with the same mechanism and ever more detailed variants:

2.2.2 Oracle

Oracle provides today very differentiated options to configure the sort and filter behavior with different mechanisms:

1. Functional index
2. Changed behavior per session
3. Case insensitive index
4. Case insensitive database collation

2.2.3 FileNet Content Manager

1. FileNet Content manager can either work with the setings on the database or
2. has an option which is called “enforce case insensitive searches” which translates queries in a way, that the searchcondition is slightly changed and a search clause is added of the type lower(search_argument) = lower(string_value)

If you are selecting option 2 of above, you are responsible to ensure the relevant functional index is available on the table. Typically, these indices should be created accordingly to following pattern including the search argument and the object_id:

```
lower(uxy_propertytemplate, object_id)
```

This allows the optimizer to choose this index instead of traversing the whole table by a full table scan.

1) Since 10gR2, Oracle allows to fine-tune the behavior of string comparisons by setting the NLS_COMP and NLS_SORT session parameters: (changed session behavior)

```
SQL> SET HEADING OFF
SQL> SELECT *
  2  FROM NLS_SESSION_PARAMETERS
  3  WHERE PARAMETER IN ('NLS_COMP', 'NLS_SORT');

NLS_SORT
BINARY

NLS_COMP
BINARY

SQL>
SQL> SELECT CASE WHEN 'abc'='ABC' THEN 1 ELSE 0 END AS GOT_MATCH
  2  FROM DUAL;

         0
SQL>
SQL> ALTER SESSION SET NLS_COMP=LINGUISTIC;

Session altered.

SQL> ALTER SESSION SET NLS_SORT=BINARY_CI;

Session altered.

SQL>
SQL> SELECT *
  2  FROM NLS_SESSION_PARAMETERS
  3  WHERE PARAMETER IN ('NLS_COMP', 'NLS_SORT');

NLS_SORT
BINARY_CI

NLS_COMP
LINGUISTIC


SQL>
SQL> SELECT CASE WHEN 'abc'='ABC' THEN 1 ELSE 0 END AS GOT_MATCH
  2  FROM DUAL;
         1
```

So this could be an approach which could simplify the documented and supported approach in the future:

```
create or replace

trigger set_nls_onlogon

AFTER LOGON ON SCHEMA

DECLARE

BEGIN

  EXECUTE IMMEDIATE 'ALTER SESSION SET NLS_SORT=''BINARY_CI''';

  EXECUTE IMMEDIATE 'ALTER SESSION SET NLS_COMP=''LINGUISTIC''';
END set_nls_logon;
```

There might be as well possibilities to overwrite the session default behavior on the database pool definition of the application server.

2) You can also create case insensitive indexes:

```
create index
   nlsci1_gen_person
on
   MY_PERSON
   (NLSSORT
      (PERSON_LAST_NAME, 'NLS_SORT=BINARY_CI')
   )
;
```

3) From later Oracle versions a case insensitive collation is available:
https://oracle-base.com/articles/12c/column-level-collation-and-case-insensitive-database-12cr2

Summary/Recommendations:
If you chose a strategy which is not documented on the IBM knowledge center, please ensure that you test your approach and that you contact your local IBM contact and ask for a blessing of your approach by the support team. This has been done in the past several times.

The case insensitive collation approach has been documented and supported on DB2
The functional indices have been documented and supported for DB2 and Oracle.

2.3 Objectstore (Content part - non workflow part)

At the creation of an Objectstore database table, indices, sequences and views are generated.
It would be a good idea to keep the DDL of a fresh created Objectstore at a safe place to be able to compare at a later time.
Over time it gets difficult to understand whether the product, a fixpack, a performance & tuning exercise or a manual dba intervention have changed database indices.
There are some recommendations around on various levels: (product, fixpacks, p&t)

Create performance enhancing indexes	https://www.ibm.com/support/knowledgecenter/SSNW2F_5.5.0/com.ibm.p8.performance.doc/p8ppt152.htm	
Indexing for IBM FileNet P8 Content Engine Searches	http://www-01.ibm.com/support/docview.wss?uid=swg21502886&wv=1	
Managing workflow indexes	https://www.ibm.com/support/knowledgecenter/en/SSNW2F_5.5.0/com.ibm.p8.ce.admin.tasks.doc/p8pcc301.htm	
DBCREATEPEVIEWS workflow system property	https://www.ibm.com/support/knowledgecenter/SSNW2F_5.5.0/com.ibm.p8.ce.admin.tasks.doc/bpfad034.htm	
Performance tuning for IBM Content Navigator	https://www.ibm.com/support/knowledgecenter/SSEUEX_3.0.8/com.ibm.installingeuc.doc/eucpt007.htm	
Setting the log size for Oracle case history databases	https://www.ibm.com/support/knowledgecenter/SSCTJ4_5.3.3/com.ibm.casemgmt.design.doc/acmtn205.htm	

2.3.1 Report existing indices per Objectstore

Following SQL Script helps to get a good understanding of the current available indices:

```
select
B.table_name, B.index_name, B.col, A.column_expression
from all_ind_expressions A,
(
select table_name , index_name, listagg(column_name, ',')
within group (order by column_position) as col
from all_ind_columns
where index_name not like 'BIN$%'
--and table_name like 'DOCVERSION'
group by table_name , index_name
order by table_name , index_name
)
B
where A.index_name (+) = B.index_name
```

DOCVERSION	AI_UDEC8_ESTV_PARTNERID_D	SYS_NC00179$	LOWER("UDEC8_ESTV_PARTNERID")
DOCVERSION	BER_HOME_ID_D	HOME_ID	
DOCVERSION	BER_RECOVERY_ITEM_ID_D	RECOVERY_ITEM_ID	
DOCVERSION	DOCTITLE_IDX	SYS_NC00173$,OBJECT_ID	LOWER("U1708_DOCUMENTTITLE")
DOCVERSION	I_DOCVERSION_DOSSIERNR	U39C6_EFIMD_DOSSIERNR,OBJECT_ID	
DOCVERSION	I_DOCVERSION_ISDOCID	UAC84_EFIMD_ISDOCID,OBJECT_ID	
DOCVERSION	I_DOCVERSION_MIMETYPE	MIME_TYPE	
DOCVERSION	I_DOCVERSION_OBJECT_CLASS_ID	OBJECT_CLASS_ID	
DOCVERSION	I_DOCVERSION_OBJ_VERS_SER_ID	OBJECT_ID,VERSION_SERIES_ID	
DOCVERSION	I_DOCVERSION22	VERSION_SERIES_ID,SYS_NC00153$,SYS_NC00154$	"MAJOR_VERSION_NUMBER"
DOCVERSION	I_DOCVERSION22	VERSION_SERIES_ID,SYS_NC00153$,SYS_NC00154$	"MINOR_VERSION_NUMBER"
DOCVERSION	I_DOCVERSION73	SECURITY_FOLDER_ID	
DOCVERSION	IDX$$_BARCODE	SYS_NC00155$,OBJECT_CLASS_ID,HOME_ID,RECOVER	LOWER("UE5E8_EFIMD_BARCODE")
DOCVERSION	IDX_CMP	OBJECT_CLASS_ID,SYS_NC00179$,OBJECT_ID	LOWER("UDEC8_ESTV_PARTNERID")
DOCVERSION	IDX_STA_1	UF176_EFIMD_FORMID,OBJECT_CLASS_ID,UB926_SOURCESYSTEM,HOME_ID,RECOVERY_ITEM_ID,CREATE_DATE	
DOCVERSION	IDX_STA_2	CREATE_DATE	
DOCVERSION	IDX_STA_3	U4838_PAGINATOR	
DOCVERSION	IDX$$_14070001	IS_CURRENT,OBJECT_CLASS_ID,HOME_ID,RECOVERY_ITEM_ID	
DOCVERSION	IPDOS_U3988_ESTV_DOK_GRUPPE_D	SYS_NC00181$	LOWER("U3988_ESTV_DOKUMENTENGRUPPE")
DOCVERSION	IPDOS_U44F8_ESTV_DOCTYPE_D	SYS_NC00180$	LOWER("U44F8_ESTV_DOCTYPE")
DOCVERSION	SYS_C008148	OBJECT_ID	
DOCVERSION	UI_U69D7_CMACMASSOCIATEDCASE	U69D7_CMACMASSOCIATEDCASE	
DOCVERSION	U8616_EFIMD_BPRID	U8616_EFIMD_BPRID	

FIGURE 1: TABLE SHOWS OUTPUT FROM THE ABOVE SQL

The above report shows the table name, the index name and the order of attributes in the index. For functional indices you see a cryptic name instead of the expression. But you get the expression in the column right to it.

From a perspective of the application, it would be great to understand which type of class has which property templates exposed and how do they be called from the perspective of the database:

```
select g.symbolic_name , c.column_name, c.dbg_table_name, c.column_datatype,
       from columndefinition c
       , globalpropertydef g
where c.prop_id = g.object_id
```

	A	B	C	D
1	SYMBOLIC_NAME	COLUMN_NAME	DBG_TABLE_NAME	COLUMN_DATAT
151	Comment	ue638_comment	DocVersion	8
152	CommunicationType	ue1d6_communicationtype	DocVersion	6
153	ComponentBindingLabel	u2e28_componentbindinglabel	DocVersion	8
154	ContainerType	u2618_containertype	Container	8
155	CustomObjectType	u26e8_customobjecttype	Generic	8
156	Description	ue0c8_description	Container	8
157	Description	ue4b8_description	DocVersion	8
158	Description	u4258_description	Event	8
159	Description	u3628_description	Generic	8
160	Description	u5c58_description	Link	8
161	DisableEH	u3fa6_disableeh	DocVersion	6
162	DocGroup	u53e6_docgroup	DocVersion	6
163	DocState	u10a6_docstate	DocVersion	6
164	DocType	ubb16_doctype	DocVersion	6
165	DocTypeVersion	u9396_doctypeversion	DocVersion	6
166	DocumentLanguage	u5446_documentlanguage	DocVersion	6
167	DocumentTitle	uc1f8_documenttitle	Annotation	8

FIGURE 2: SEARCHING A LIST OF SYMBOLIC_NAMES, IT'S COLUMN NAME AND THE RELATED DATABASE TABLE

The report above shows multiple attributes (property_templates) which are used in different contexts.
Document Classes are mapped into the table "DocVersion", Folders and Case Folders are mapped in "Container", Custom Objects in "Generic", RootClasses in "UT_..." etc.

Regardless whether you have one or multiple document classes, all are sharing a huge database table "Docversion".
Saying this, the table may contain many attributes and for search reasons you need multiple indices.
So it makes sense to minimize the number of database indices if possible to reduce the overhead.

2.4 Objectstore - Workflow System part

With the usage of Workflows and in particular with Case Manager the combination of both area (content and workflow) must be tuned as well
The workflow system follows a bit a different approach.

How to gather the mapping between Roster, Queues and EventLogs and the Database Views: Use vwtool (see
https://www.ibm.com/support/knowledgecenter/en/SSNW2F_5.5.0/com.ibm.p8.pe.vw.doc/bpfvl051.htm

For tracing the JDBC calls against the VW-database objects, use as well vwtool
https://www.ibm.com/support/knowledgecenter/en/SSNW2F_5.5.0/com.ibm.p8.pe.vw.doc/bpfvl049.htm

Switch on:

Database access	This option outputs the SQL statements that are used by workflow system, along with the values of the substitution variables. Setting this option automatically sets the database time option.
Database outputs	Trace the field values and outputs of database statements, such as SELECT, UPDATE, and so on.
Database time	This prompt appears only if you answer no to the database access prompt.

Typically, the workflow system out of the box is already set reasonably well leveraging database indexing. You should mainly check whether you are applying heavily filters and try to avoid designs where one queue is holding millions of work items.

There is no way to run case insensitive searches using the Workflow system other than what has been configured on the database side. Typically, a Workflow System is using status meta data and is typically not really sensitive to cases. There is no such feature like on the content system where the queries would apply a lower() function.

If you are considering using the workflow subsystem, maybe case management then think about an approach which makes the database case insensitive!

3 Creation of repository artefacts

Content Management and in particular Case Management will requirement multiple metadata.
On customer projects I have experienced that the generation of such meta data could be done using the tools on the Objectstore or using the implicit generation when using Case Builder as your tool. So it would be great to have a common view onto the Objectstore regardless how a meta data field has been created.

3.1 Generating artefacts

An Objectstore contains many property templates and many classes.

Property Templates can be created by:

- Manually in ACCE
- Case Builder
- By product installations
- By addons
- Deployment Manager
- Configuration Manager jobs
- P8Toolkit as part of manual jobs
- P8Toolkit as part of UCD deployments

It is a good idea to use the category system field to tag the various artefacts for its purpose.
There should be a document describing the data model from a solution perspective. It is mainly important to manage the dependencies between artefacts being created outside of CaseBuilder and in CaseBuilder. Translation might be still a challenge from within CaseBuilder.

3.2 Stable GUIDs

There is a concept, that every artefact should have the same global identifier in each stage so that dependent artefacts like folders, searches etc. are still referring to the correct artefacts. This simplifies the situation when transporting artefacts from stage to stage, it is imperative to use the same IDs. In particular this is important when creating an Objectstore using the tools instead if the API, where you could change the GUID at creation time.

FileNet Deployment Manager is using exactly this concept to transport the artefacts between stages.

3.3 Configuring Search behavior

In the screenshot below you can see how to set the behavior of case insensitive searches using ACCE:

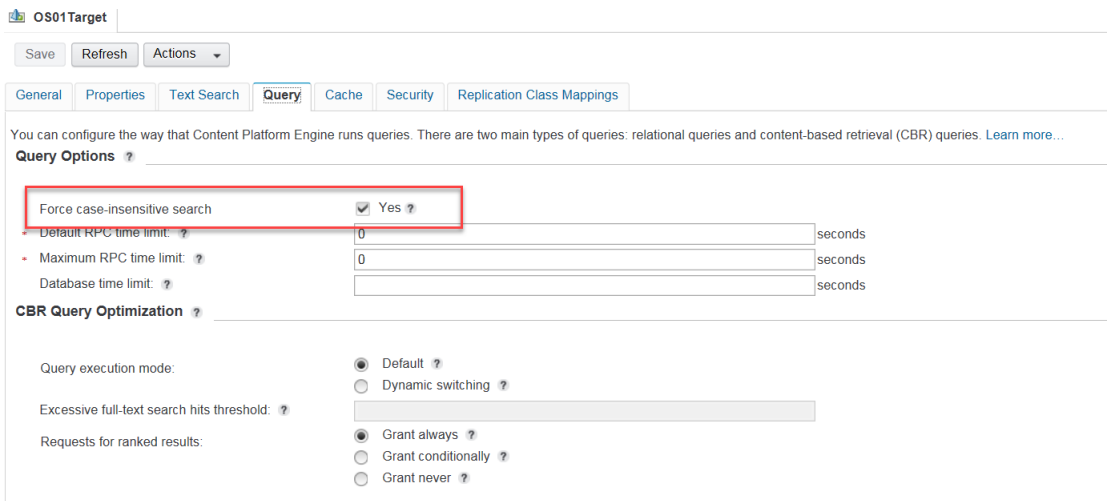


FIGURE 3: CASE INSENSITIVE SEARCHES

This means, all string comparisons are formulated by the Content Platform Engine as `lower(search_argument) = lower(string_value)`
This implies that there are functional database indices created of the form `lower(uxy_propertytemplate)`.
There is a dependency between the symbolic_name and the implemented database column name which may vary between stages (dev/test/…prod). This enforces us to work with symbolic_names in logical index creation scripts which can then be leveraged to produce stage depended SQL-ddl files for the index creations

The database column name can be either queried from the Objectstores repository or through the API.

For people who are interested I can share a utility which allows to create the DDL for index creation based on symbolic names of the property templates and the symbolic name of the class.

e.g.

```
create index #DBSCHEME#DBSCHEME#.I_docv_docdate on
#DBSCHEME#DBSCHEME#.docversion(##DocClass#,object_id)@
```

will get translated into:

```
create index obj1.I_docv_docdate on
obj1.docversion(u0ec8_subject,object_id )@
```

4 Queries

Queries can be executed from:

- ACCE (Adhoc or saved)
- custom programs
- search templates in Navigator
- ..

It is often hard to understand who is executing what kind of query. As a best practice following recommendations can be made:

- Document every custom query
- Do not embed queries in Java Code but externalize the queries and define a queryID.
- Mark the query technically:

```
AND creator <> 'QueryId: Query123'
```

The attribute creator is available on all kind of objects like folders, document classes and custom objects

The additional where clause is not changing the query and the optimizer should not be irritated because the creator is not really a very selective attribute.

```
SELECT [This], [DocumentTitle], [DocState], [EAGSNR], [DateCreated],
[MajorVersionNumber], [MimeType], [Id] FROM [DOCCLASSX] WHERE [DossierNr] =
959614 AND [IsCurrentVersion] = true AND [EAGSNR] is not null

AND creator <> 'QueryId: Query123'

ORDER BY DateCreated ASC
```

Adapt all Custom CE Queries and apply such behavior with the creator clause and a queryId

5 Testing – Single User Tests

Whenever a query from a use case is ready for testing, allow some time on a FileNet Content Manager server where there is no additional traffic from other users. This means, try to get exclusive access to this machine and switch on tracing.
In a multi node cluster, stop all but one server to simplify searching the relevant trace from only 1 and not from all possible log files.
If you know that you can generate a queryID which is absolutely unique, it may as well work in an environment where you are not granted exclusive access to the CPE server.

5.1 Instrumentation

5.1.1 CE Traces

With the instrumentation of the CE Traces, it is possible to show activities on various interfaces. Typically, only the database layer is of relevance.

Tracing can be switched on according to following documentation
http://www-01.ibm.com/support/docview.wss?uid=swg21308282
The subsystems which might be useful are:
https://www.ibm.com/support/knowledgecenter/en/SSNW2F_5.5.0/com.ibm.p8.common.admin.doc/logs/cpe_logging_concepts.htm

- DatabaseTraceFlags (this shows the jdbc sql, the inbindings and the execution time)
- SearchTraceFlags (this shows the notation which can be used in ACCE fro replays)

5.1.2 Database Traces

With the instrumentation of the database it is much easier to understand the effective costs.
With the trick of applying a binding variable creator and its value, it is fairly easy for the DBA to search for a particular query. In cases where this is not possible the CE Trace will include the SQL syntax of the database. The dba can search for the query text.
For details consult the Oracle dba.

5.2 Analysis

Document the use case, the CE trace and the execution plan from the database.

5.2.1 Analysis of the CE Trace

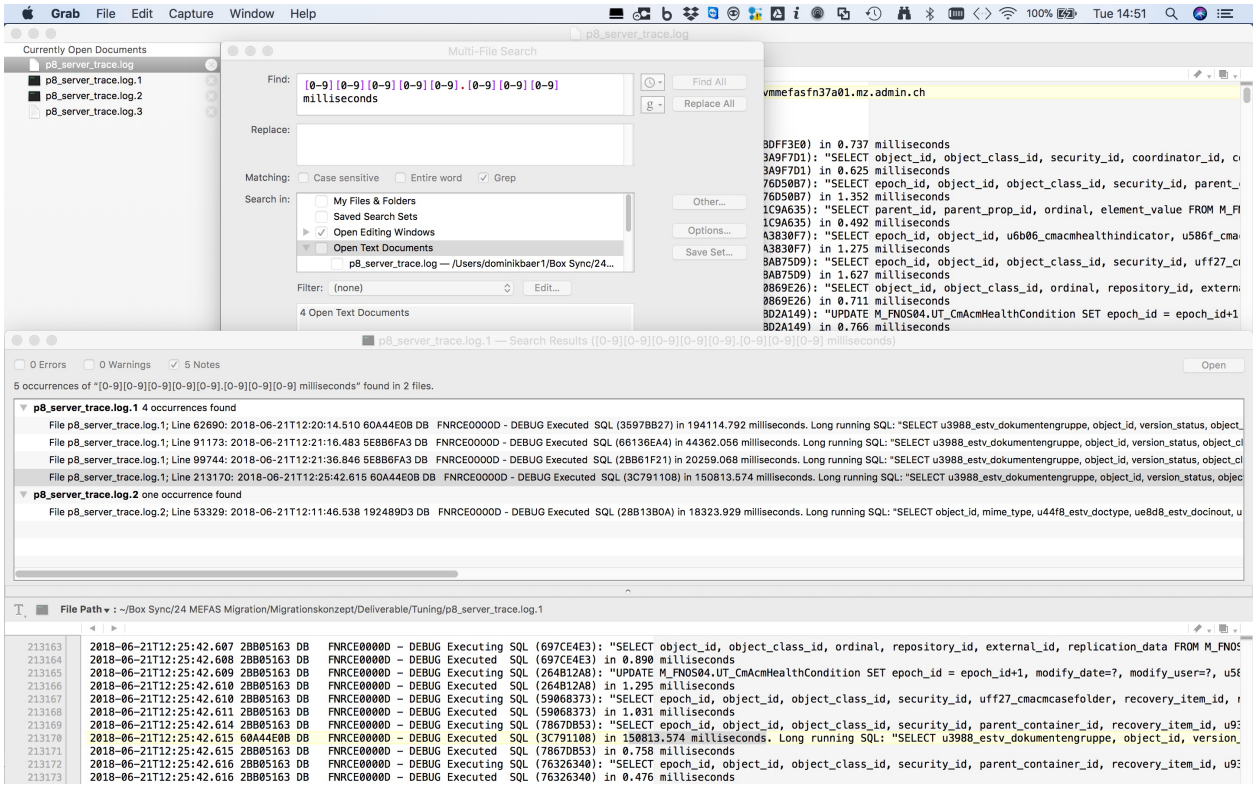


FIGURE 4: ANALYSIS OF CE-TRACES

With the help of notepad++ (Windows) or Textwrangler (Mac) the trace files can be easily searched for the queryID or for long running queries. Depending on what you are looking for a search pattern like

`[0-9][0-9][0-9][0-9][0-9].[0-9][0-9][0-9] milliseconds` might be useful.

5.3 Tuning

For every query which results in a full table scan, consider creating an appropriate database index.

Allow more time to build up a query inventory and document the query behavior according to the suggested points in this chapter.

6 Testing - Mass Tests / Loadtests

6.1 Instrumentation

Dba will extract a list of expensive queries.

Multiple long running SQL Queries can be found as well by some monitors.

DB_NAME	Applikation	ORDER	USER_NAM	ELAPSED_SEC	EXECUTIONS	Executiontime per run	Not acceptable	DISK_READ	CPU_SEC	PHYSICAL_READ_MB	PHYSICAL_	SQL_ID	DETAILS	SQL_TEXT
PFITFN05		ELAPSED		1440	20'573'505.00	0.000069992935	-	1	658	0.01	0	24u9n48b8x	Detail	SELECT T0.
PFITFN05		ELAPSED		2397	14'074'198.00	0.000170311658	-	9271	1072	72.44	0	c0p9qa3nva	Detail	SELECT epo
PFITFN05		ELAPSED		1574	1'243'674.00	0.001265604974	-	226870	231	1438.41	0	3bk9z2z8by	Detail	INSERT INT
PFITFN05		ELAPSED		621	544'814.00	0.001139838550	-	254140	86	1985.54	0	3xmbdqh1j9	Detail	SELECT epo
PFITFN05		ELAPSED		1198	392'890.00	0.003049199521	-	384	602	3	0	bvun6tq7qnx	Detail	SELECT par
PFITFN05		ELAPSED		524	66'424.00	0.007888714922	-	83544	27	652.31	0	ghuqkw9fxd	Detail	INSERT INT
PFITFN05		ELAPSED		625	16'560.00	0.037741545894	-	102397	165	776.42	312.98	g49u1uyu17	Detail	INSERT INT
PFITFN05		ELAPSED		1434	13'810.00	0.103837798697	-	150331	376	1174.46	1004.56	a8qw766fpn	Detail	INSERT INT
PFITFN05		ELAPSED		4030	7'837.00	0.514227382927	1	138	1	1.08	0	a9uanxffv05	Detail	SELECT last
PFITFN05		ELAPSED		2459	4'168.00	0.589971209213	1	6	1348	0.05	0	gx9ujkacxp1	Detail	SELECT epo
PFITFN05		ELAPSED		596	2'541.00	0.234553325462	-	195350	19	1526.17	0	20xwwgfx2g	Detail	SELECT epo
PFITFN05		ELAPSED		1014	1'725.00	0.587826086957	1	5	546	0.04	0	fm9z8bbq21	Detail	SELECT obj
PFITFN05		ELAPSED		2384	667.00	3.574212893553	1	274712	83	2146.19	0	1yzn5rss1m	Detail	INSERT INT
PFITFN05		ELAPSED		442	254.00	1.740157480315	1	52426	25	0	0	614d2z99b0	Detail	UPDATE M_
PFITFN05		ELAPSED		648	21.00	30.857142857143	1	2	46	6748.02	0	8zyjczrdnpa	Detail	select /*NOR

FIGURE 5: TOP SQL, LONGRUNNING SQL STATEMENTS

Long running SQL Queries can be found with various database tools. Figure above shows the `elapsed_sec` and `Physical_READ_MB`. From these two key performance indicators the average execution time and the average number of physical reads per execution can being derived.

As a rule of thumb execution time higher than 500 – 1000ms are candidates for long running queries. Depending on its frequency they should be optimized.

From a perspective of I/O physical reads greater than multiple 100 MB are as well candidates for slow performers.

Regarding documents being stored in the database. If documents are stored in the database, a query which is delivering content must be identified from its SQL_TEXT. It reads from a table “content”. The high costs in physical reads are typically being produced by tables scans instead of index seek or index range scans.

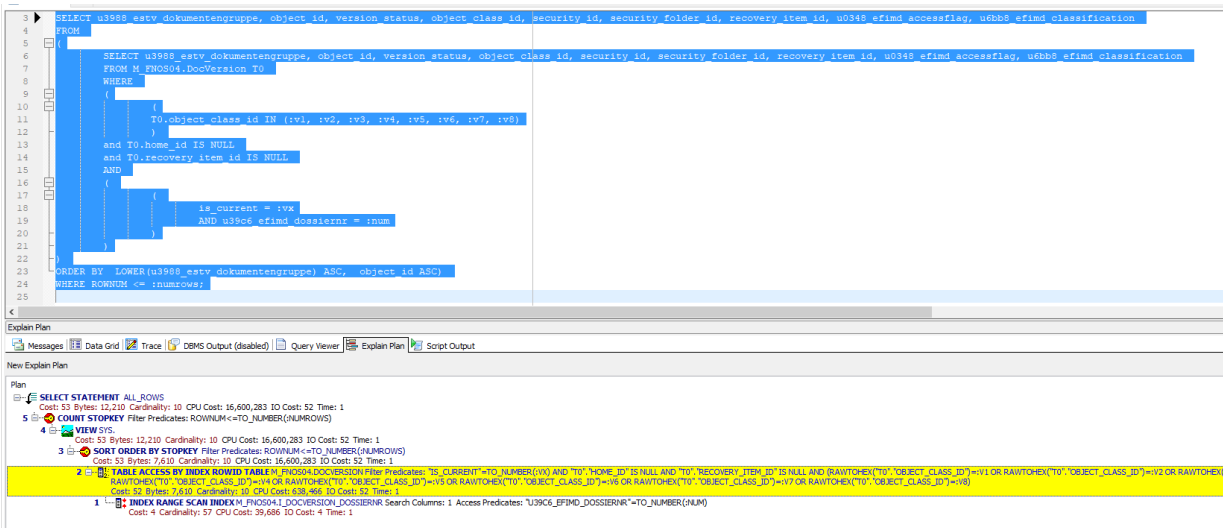
6.2 Analysis

The extracted text must be manually reformatted resp. toad has such editor capabilities

```
FNRCE0000D - DEBUG Executed  SQL (3597BB27) in 194114.792 milliseconds. Long running SQL: "
SELECT u3988_dokumentengruppe, object_id, version_status, object_class_id, security_id, security_folder_id,
recovery_item_id, u0348_accessflag, u6bb8_classification FROM (SELECT         u3988_dokumentengruppe, object_id,
version_status, object_class_id, security_id, security_folder_id, recovery_item_id, u0348_accessflag,
u6bb8_classification
FROM M_FNOS04.DocVersion T0
WHERE
(
                        (
                        T0.object_class_id IN (?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?)
                        )
        and T0.home_id IS NULL
        and T0.recovery_item_id IS NULL
        AND
        (
                (
                         is_current = ?
                        AND LOWER(u44f8_doctype) <> LOWER(?)
                        AND
                        (
                                LOWER(udec8_partnerid) = LOWER(?)
                                OR u69d7_cmacmassociatedcase IN
                                        (
                                                SELECT  T1.object_id FROM M_FNOS04.Container T1
                                                WHERE
                                                 (
                                                        (
                                                                T1.object_class_id IN (?, ?, ?, ?, ?)
                                                        )
                                                        and T1.home_id IS NULL
                                                        and T1.recovery_item_id IS NULL
                                                        AND
                                                        (
                                                                LOWER(T1.udad8_estv_partnerid) = LOWER(?)
                                                        )
                                                )
                                        )
                        )
                )
        )
)
ORDER BY  LOWER(u3988_estv_dokumentengruppe) ASC,  object_id ASC)
WHERE ROWNUM <= ?
" In-bindings: ({4EADED59-2170-4606-91F5-704C4653D373},{809DBBC2-B032-4F6C-B697-AECCF9F8EDF9},{9640BF23-1B7E-4C2F-
8B1A-FF213D619E32},{BADAE424-9DF4-42DC-980E-B00293645C6C},{0DDF01DA-DEF9-4639-9C72-B9268A0B5745},{22E3FF1D-6A0A-
4F0B-88D9-949172939A9C},{BBF92363-945B-427E-BBD9-E636F31FC511},{223AA027-CA30-4E86-9395-E6E2D0C46B3E},{1B3F1C5E-
C5FD-40F9-A5F3-5585F2F3CEA0},{B242F976-CE4B-4576-A0BC-293F3A433029},{72C6E23F-2AA8-48E4-9583-
93C3409F8F10},{B5B06FA1-BBB3-461C-9243-946D9BFE0720},{3C496105-D4EC-4463-B8EE-70957F50F0E2},{00151BA6-E80E-4C6F-
8E4F-8778550163CF},{F40BBAD5-7A97-4BFF-A70D-60D588A6FCB4},{904EAEE4-01F7-47E8-89EE-D6C971E81860},{C9776F9B-1AA9-
4226-AED7-755577D9E3EA},{55904B14-44DD-4D47-8EAC-4FC9B8677790},{D6872636-EA8B-42E2-9B76-F79C67B6B5FE},{804D49AC-
8508-4CE9-B792-ECB4C6A0E798},{B03AD059-0000-CB18-8EF4-A4CFD4071C38},true,'Request','5202238491',{0B0CD12E-23B4-
4E09-B7DD-8D63B910C5EE},{79AAA9BA-C744-4840-AE0A-96D2CE35EBE8},{E7DF4579-0E65-4A8C-9AD3-FA93B32C3606},{758DD48F-
78A9-4B32-B047-3C55CF0D095C},{3D269935-0ADE-4C59-81D7-9CD5FEE3EFEA},'5202238491',1000)
2018-06-21T12:20:14.510 2BB05163 DB
```

This gives a better overview and allows to understand what kind of where clauses and order by statements are generating a certain cost.

With the database tools it is possible to generate a graphical representation of the execution plan:



Monitor the performance on the productive database monthly and take actions.

7 Documentation

- Document all searches from a logical point of view (user, application, …)
- Map them to search templates or to applications and its code.
- If possible do not create the queries in the Java code but externalize the SQL (PseudoSQL) in a textfile which allows the tuning independendly from a release/deployment.
- Add the additional filter condition with a query identifier
- Run the query using ACCE and document the execution plan on the database.
- Document all indices on the relevant tables and save the DDL for these as part of the release/deployment package
- Decide which property templates are not used anymore. Get rid of these indices.

7.1 Table - Docversion

Index on documenttitle was created before the deployment.

CREATE INDEX M_FNOS04.idx_cmp ON M_FNOS04.docversion (object_class_id, LOWER(udec8_partnerid), object_id) tablespace TS_FNOS04_INDEX compress 2 parallel 4 nologging;
alter index M_FNOS04.idx_cmp noparallel logging;

7.2 Table - Container

CREATE INDEX "M_FNOS04"."AI_UDEC8_PARTNERID_C_O" ON "M_FNOS04"."CONTAINER" (LOWER("UDAD8_PARTNERID"), object_id) tablespace "TS_FNOS04_INDEX" compress 2 parallel 4 nologging;
alter index "M_FNOS04"."AI_UDEC8_ESTV_PARTNERID_C_O" noparallel logging;

7.3 Table - Task

create index "M_FNOS04"."U8B88_ESTV_INFOCARRIERI" on "M_FNOS04"."TASK" (lower(U8B88_ESTV_INFOCARRIERINSTANZI), object_id) tablespace "TS_FNOS04_INDEX" compress 2 parallel 4 nologging;
alter index "M_FNOS04"."U8B88_ESTV_INFOCARRIERI" noparallel logging;