**TD STECH DB Architecture**

* Main business in TD STECH is distribution. We have ERP system to support business, which we call CIS system.
* Here is the Main Database Servers

**OLTP OLAP**

**UK PROD**

**OLTP OLAP**

**HYCN PROD**

**OLTP OLAP**

**XD PROD**

**OLTP OLAP**

**MX PROD**

**OLTP OLAP**

**HYVE US**

**PROD**

**OLTP**

**US CORP**

**OLTP**

**CA CORP**

**OLAP**

**DWS**

**OLAP**

**CA DWS**

* US/CA/UK/CN/MX Represents different regions and different servers. XD is Xerox company.
* US has two servers, same as CA. One is US CORP for OLTP, the other is US DWS for OLAP, with replication to build the connection.

(OLTP Stands for online transaction process. OLAP stands for online analytical process. DWS stands for Data Warehouse System (OLAP)).

OLTP:US CORP/CA CORP || OLAP:US DWS/CA DWS

* HYVE US PROD/UK PROD/HYCN PROD/MX PROD are 4 independent servers for 4 different regions. XD PROD is for xerox company. They 5 don't have severs to be separated to 2 servers like US and CA. They only have one server named PROD. But PROD still has both process of OLAP and OLTP. In PROD OLTP also use replication to build connection with OLAP. You can understand like that OLAP & OLTP (CORP and DWS) are in the same server named PROD).

**TD STECH CORP (US CORP)**

**CORP**

**OLTP**

CIS

TD SYNNEX business data

INT

Archive

HIS

* CORP is OLTP
* Here is main database in CORP and DWS
* CIS is read-only for DWS.CIS for CORP is from kinds of business support system, which is not read-only.
* HIS is archive of CIS.
* INT is temporary database. Data will not change when database server restarts.
* We have the other temporary database named tempdb. The difference is that tables in tempdb will lose data or be dropped when server restarts.

**TD STECH DWS from CORP (US CORP)**

* DWS from CORP
* Replication Frequency
* Real time for CIS.
* Weekly for HIS.

**DWS**

**OLTP**

**OLTP**

**CORP**

**DWS**

**CORP**

**Sync weekly on weekends**

**Nightly**

**loading**

Archive

**Real-time Replication**

INT

INT

**DW\_PROD**

HIS

HIS

CIS

CIS

**TD STECH CORP and DWS (US)**

* **TD STECH DWS**
* DWS is OLAP
* CIS for DWS is read-only.
* DW\_PROD is its unique database.
* Data in DW\_PROD is based on data of CIS through cron(Stored procedures and shell scripts)
* Data in DW\_PROD can be modified.

**DWS**

**OLAP**

**CIS**

**B**

**report**

**INT**

**Nightly loading**

**Data Source**

**Other**

**Application**

**report**

**CRON report**

**RDS report**

**DW\_PROD**

**HIS**

**DWS**

**DBA LIMIT**

5M/ I/O with kill

Over 40% on tempdb for single process

Log segment thresholds: 40%

5-min long blocking

2-min long blocking

1M/ I/O with kill

Over 40% on tempdb for single process

Log segment thresholds: 40%

5-min long tran

2-min long blocking

**CORP**

**DWS**

* the space of the whole logical log is relatively fixed and cannot grow unlimited.

The use of logical log files is circular. If log can't be released for reuse, then system will hang.

So Log segment thresholds can't exceed 40% from DBA's request for performance.

* The tempdb usage of a process exceed 40% of total tempdb size is not allowed.
* If the sql execution's I/O already read 5M/1M, it will be killed by db safe guard.

**Major SQL Case Type**

* Major SQL Case Type From 12/1/2012 to 12/1/2014

|  |  |
| --- | --- |
| Category Type | Count of SQL Cases |
| Blocking | 115 |
| IO | 153 |
| Long Run ?-execution time over 4 minutes | 59 |
| Long Tran | 153 |
| Open C open transaction | 18 |
| Rowcount | 20 |
| TableScan | 160 |
| tempdbUsage | 69 |
| DataQuality | 1 |
| DataLifeCycle | 1 |
| DDL tune(not sure about the name) | 6 |
| Data Integrity – Orphan Data Checking | 18 |
| Data Integrity – Null value in Unique Indexes | 17 |
| High CPU usage | 195 |

**TD STECH DEV and UAT**

* DEV and UAT from CORP & DWS
* DEV server is a development database
  + On server dev, developer has the rights of database administrator(dbo)
* UAT server is a test database of TD SYNNEX
  + It is a semi product database. User has all permissions except creating objects.

**UAT**

**DEV**

**Sync Weekly at weekends**

**Sync Weekly at weekends**

**DWS**

**CORP**

**DW\_PROD**

INT

**Sync weekly on weekends**

CIS

HIS

Archive

CIS

**DW\_PROD**

HIS

**TD STECH Database object deployment process**

**One Tool Process**

**Design and preparation**

**DB Server**

**Rule**

* The process is mainly aimed at creating table and stored procedure SP
* Yellow is rule
* Green is database
* Blue is deployment readiness
* Pink is deployment process
* How to know data delay between primary DB and replicate DB?

exec CIS..reptime;

Create DP (Deployment Plan) based on backlog or release.

Confirm the need to create a table/Business data or report data/if must create table on CIS

If the old SP is not formatted, the first step is to format old SP and submit to CSV. Then add new changes and submit again.

Create DO Submit tested cod and apply

Code and test on DEV

Format code with Notepad++ and prepare DDL code

Fill DMR information, which is only valid for the table.

Integrity Review

Deploy to UAT by one Tools.

DM Review

Fill DMR information

Deployment time window: Monday, Tuesday, Thursday, the last week of each month cannot be deployed, and the 25th of each month (the billing day) is blackout.

IT Leader Approve

QC and Exec Approve

Deploy

**Create table in STECH DB**

Judge before deleting objects

IF object id('sample\_details ') is not null

drop table sample\_details

go

CREATE TABLE sample\_details(

id int identity

, sample\_no int not NULL

,sample\_name varchar(60) NULL

,sku\_no int not NULL

,scm\_no int not NULL

,rebate\_amt money not NULL

,begin\_date datetime NULL

,expire\_date datetime NULL

,approved\_cost money NULL

,active char(1) not NULL

,close\_date datetime NULL

,close\_id int NULL

,entry\_datetime datetime not NULL

,entry\_id int not NULL

,min\_cust\_qty int NULL

)WITH identity\_gap = 500

GO

CREATE UNIQUE CLUSTERED INDEX sample\_details**I1** ON sample\_details (sample\_no ASC,sku\_no ASC)

GO

CREATE INDEX spa\_detail**I3** ON sample\_details(scm\_no ASC)

GO

CREATE INDEX spa\_detail**I2** ON sample\_details2(sku\_no ASC, active, close\_date ASC, sample\_no ASC)

GO

GRANT ALL

ON sample\_details

TO public

GO

* Start with letter
* It is separated by "\_" and consists of lowercase letters and numbers
* Table name cannot exceed 27 characters
* Create a fixed temporary table in INT
* Include timestamp (entry date) and operation source (entry id)

In order to avoid the jump of identity column, the gap value should be set

* Index must be named by TablenameI**[1,2,3]**
* The first index uses business fields/field combinations that represent the uniqueness of the table as much as possible
* Identity field can't be index field
* Long strings (>50) cannot be used as keys for indexes.
* One table only has one clustered index.

93%

Long strings (> 50) cannot be used as keys for indexes

One table only has one clustered index.

**Grant to public**

**Naming specification-1 in STECH DB**

* The object name in database should be as short and meaningful as possible
* The object name in database is separated by '\_' without space
* The object names in database are all in lowercase
* Avoid using keywords for object names in databases

select name from master..spt values where type='W' order by name

* The length of table name cannot exceed 27 characters

|  |  |
| --- | --- |
| Right Case | Wrong Case |
| 1. order\_header | 1. order&header 2. order\_ header 3. order\_headerxxxxxxxxxxxxxxxxxxxxxxxx   xxxxxxxxxxxxxxxx (length>27)   1. Order\_header |

* Use appropriate terms and abbreviations: (You can review our tables in CIS to know the columns' naming specifications better)

1. Use unified writing specifications, such as "cust", 'vend", etc. (need to define abbreviation list.)

1. For field names with the same purpose, try to use the same name, type and length in the same database; For example, **entry\_datetime,entry id, date flag,zip code**, etc. (need to define the list

|  |  |
| --- | --- |
| Right Case | Wrong Case |
| 1. vend\_no 2. entry\_datetime 3. zip\_code 4. cust\_no | 1. vendor\_nc 2. entry\_date, entrydatetime… 3. zipcode 4. custno, customer\_no |

**Don’t forget to format before submitting**

* How to find the relevant specification fields
  + Refer to the relevant tables through sysobjects & syscolumns
* Demonstrate a SQL
  + Requirements: extract the related fields and field types of an existing database table and whether they are empty

select a name+' '

+case a usertype when 7 then 'int'

when 6 then 'smallint'

when 2 then 'varchar'

when 11 then 'money'

when 1 then 'char'

when 8 then 'float'

when 12 then 'datetime'

WHEN 19 then 'text'

else 'other' end

+case when a usertype in(2,1,8) then '('+convert(varchar(4), a length)+')' end +"

+case when status=0 then 'not NULL' else 'NULL' end

from **CIS..syscolumns** a join **CIS..sysobjects** b on a.id=b.id

where b name='customer header’

order by a colid

* For newly created fields, **smallint is not allowed**. Instead, int should be used
* Add "grant all on xxx to public" to the newly created table
* Gap must be defined in the Identity field, and the value depends on the specific situation (10-10000). The more frequent the table operation, the larger the defined Gap will be.

|  |  |
| --- | --- |
| Right Case | Wrong Case |
| Create table xxx(sid int identity, ….) with identity\_gap=10 | Create table xxx(sid int identity, ….) |

**Index**

* Index must be named by Tablenamel [1,2,3...]

|  |  |
| --- | --- |
| Right Case | Wrong Case |
| dw\_orders**I1** | idx1\_dw\_orders… |

**create clustered index xxxI1 on xxx()**

* The first index should be the **[unique] [clustered] index** of the table
* The first index uses business fields/field combinations that represent the uniqueness of the table as much as possible
* Table in CIS must have unique index
* **Long strings (> 50) cannot be used as keys for indexes**

**Rules for stored procedures**

* The structure of a stored procedures in STECH:

Must switch to the database where the stored procedure is located

use CIS

GO

IF object id consign create orders') IS not NULL

BEGIN

DROP PROCEDURE consign create orders

If object id('consign create orders) IS not NULL

print '<<< failed dropping procedure consign create orders >>>

ELSE

Judgment and prompt information format

print '<<< dropped procedure consign create orders >>>

END

GO

CREATE PROCEDURE consign create orders

WITH recompile

Start with ‘AS’ End with ‘GO’

AS

/\*\*

\*Deployment information

\*Modified information

\*/

DECLARE @lol\_account int

Variable definition (@)

@usend int

@row int

-create temp table for batch insert order detail

IF object id #temp ord") IS not NULL

BEGIN

Truncate before dropping a temporary table. Do not use delete operation.

TRUNCATE TABLE #temp\_ord

DROP TABLE #temp\_ord

END

**Rules for stored procedures**

CREATE TABLE #temp\_ord(field data\_type

….

WHILE (@@rowcount > 0)

BEGIN

--insert order header

BEGIN TRAN

IF (@is\_new\_order = "Y")

BEGIN

**@@rowcount is common system variable that returns the number of rows affected by the previous statement**

….

SELECT @row = @@rowcount

IF @row = 0

BEGIN

print "Problem in insert order\_header"

ROLLBACK

END

END

ELSE

BEGIN

….

**Open transactions are allowed**

END

--insert order detail

--get order line no

COMMIT TRAN

….

END

**Rules for stored procedures**

**You may also select a result set**

select (column list) from #temp order by spa\_no,sku\_no

truncate table #temp\_spa

drop table #temp\_spa

**End of procedure**

GO

IF object\_id('#temp\_order\_details') IS not NULL

BEGIN

**Truncate table before dropping temporary table**

TRUNCATE TABLE #temp\_order\_details

DROP TABLE #temp\_order\_details

END

GO

**Grant to Public**

GRANT EXECUTE

**Set mode to anymode if needed to be called in other processes**

ON consign\_create\_orders

TO public

EXEC sp\_procxmode 'consign\_create\_orders

**Do not have a space before go. Or it will report an error.**

"anymode

GO

IF object\_id('consign\_create\_orders) IS not NULL

print '<<< created procedure consign\_create\_orders >>>

ELSE

**Print related message**

print <<< failed creating procedure consign\_create\_orders >>>

GO

**Delete data of 6 months ago in CIS DB after inserted into HIS DB**

**Unified time input**

declare @stmonth datetime @rows int.@now datetime. @err no int

select @sixmonth = convert(char(11),dateadd(month, -6, getdate()). 1)

**begin tran**

* **Last SQL rows**
* **System time at the moment**
* **System error code**

insert HIS history carton header

select from carton header

where or dttm < @sixmonth

select @rows = @@rowcount, @now = getdate().@err\_no=@@error

print "STATUS: Inserted %1! rows in HIS history carton header at %21", @rows, @now

**Rollback when encountering error and print message**

if @err\_no != 0

**begin**

rollback tran

print "STATUS: error %1! when inserting HIS. history carton header @err\_no

**end**

else

**Gets information about the delete operation**

**begin**

delete carton header where or dttm < @sixmonth

select @rows = @@rowcount. @now = getdate().@err\_no-@@error

print "STATUS: Deleted %1! rows in carton, header at %21", @rows, @now

if @err\_no != 0

**Rollback when encountering error ad print message**

**begin**

rollback tran

print "STATUS, error %11 when deleting carton header @err no

**end**

**Don’t forget to commit tran after the transaction ends!!!**

**end**

**commit tran**

go

* **@@rowcount returns the number of rows affected by the previous statement**
  + Print out the number of lines affected
  + print '%1!, rowocunt = %2!',@st,**@@rowcount**
* **Rowcount can be used to control the loop (pls test on dev or uat)**
  + Due to 10 constraints, we need to control the number of rows per update

declare @a int

**Only update 5000 rows at a time**

set @a = 5000

**SET rowcount 5000**

while @a=5000

begin

**Don’t forget to commit tran after the transaction ends!!!**

update tempdb.rds\_mid\_633 set **flag**='Y'

**where flag is null**

**set @a = @@rowcount**

end

**Don’t forget to commit tran after the transaction ends!!!**

SET rowcount 0

Go

**Standard**

* **Follow SQL92**
* Explicitly use **inner join, left join, right join,** not exists out full join in SYBASE (use left join union right join)
* **dbo.** is not allowed in DDL scripts.
* **Use create table xxx;** **insert xxx; instead of select… into xxx from xxxx**

|  |  |
| --- | --- |
| **Right Case** | **Wrong Case** |
| Create table #temp (order \_no int null)  Insert #temp select order\_no from CIS..order\_header | Select order\_no into #temp from CIS..order\_header |

* **Avoid using select \* in a select statement, specify column names instead**

|  |  |
| --- | --- |
| **Right Case** | **Wrong Case** |
| Select order\_no, order\_type from #temp | Select \* from #temp |

* Don’t create and drop the temp table in loop!

**Emphasize the standard**

* **The following questions will be rejected by reviewer / IT leader/IIC/IT exec**
* SQL92 is not used (wrong)(slide 21)

**Wrong case:** select order\_no from order\_header oh,order\_detail od where oh.order\_no-od.order\_no

**Right case:** select order\_no from order\_header oh inner join order\_detail od on oh.order\_no=od.order\_no

* When Table A is updated and multiple tables are associated, table A is not immediately after "from 8, A"

**Wrong case:** update order\_header set order\_no=1 From order\_detail od inner join order\_header oh on oh.order\_no=od.order\_no

**Right case:** update order\_header set order\_no=1 From order\_header oh inner join order\_detail od on oh.order\_no=od.order\_no

Use "if a=null" when making judgments of Null (wrong)

**Wrong case:** If a = null set b =null

**Right case:** if a is null. When judge, use (is null). If a is null set b =null

Use select \* or select \* into... (wrong) (refer to slide 21)

**Right way:** select fileld1, filed2...

* Tran end without commit tran(wrong)(refer to slide 17)
* No formatting of your code (tool Notepad++) (wrong) (refer to slide 9

Format before you submit your code on one tool. It's requested and please obey this rule.

* When select has a group by or aggregate, **Result set** is not filtered in the where condition or ‘on' the join condition. (wrong) (Pls refer to slide 24)

**Right way:** **Result set** filter by **having**

Where clause filters rows. Having clause fliters result set

**Groups and Aggregates in SQL**

* SQL. handles groups and aggregates by:
* Selecting the data into a work table
* Sorting and/or aggregating the data
* The Where clause filters the data going into the work table
* The Having clause filters data retrieved from the work table

**Basic part of optimization**

* Please sort the execution order of following SQL:
* Select, from, where, group by, having, order by

**having**

**select**

**Group by**

**where**

**Order by**

**from**

**Data load in STECH DB**

* BCP operation is faster when a large number of data import and export database
* Set to 1000 lines one Tran: -b1000
* the client database is installed locally. Using the command page, you can directly access the database to import or export data
* Case
* Import data from a folder named out to the established table tempdb..sku

**bcp** tempdb..sku **in** d:\out\sku.txt -SDWS -Uuser-Ppassword-c-b1000

* Export data in the database to ser.txt under folder out

**bcp** tempdb.. serial **out** d:\out\ser.txt-SDWS-Uuser-Ppassword-c-b1000

**Basic part of optimization**

* **View execution plan**

To determine which access method the Query Optimizer has chosen, use showplan

**set showplan on(off)** (show plan and execute sg!)

**set showplan, noexec on(off)** (show plan and don't execute any sql)

**set showplan, fmtonly on(off**) (show plan and execute sql, But don't get data of result. For select type, only get the columns of the result]

* **View** **consumption of disk read and write time**

set statistics to on/off)

set statistics time on(off)

* **Three kinds of data access methods in Sybase**
* Table Scan
* Clustered Index Access
* Nonclustered Index Access
* **Covered Query**

A covered query retrieves data from the index without accessing the table

**Basic part of optimization**

**Table Scan**

* In a table scan, ASE accesses the data sequentially:
  + All pages in the table are accessed
* A table scan is chosen when:
  + Most or all of the data in the table is to be accessed
  + There are no indexes
  + There are no useful indexes
  + select from authors

**Clustered Index Access**

* A clustered index is a balanced tree structure that:
  + Forces the data into sort order
  + Provides random access to the data

Select \* from authors

where au id = '918-44-9251’

**Using a Clustered Index**

* A clustered index is chosen:
  + When there is a Where clause that references the column(s) in the index
  + When the cost of using the clustered index is less than other access methods
* Clustered indexes are most effective:
  + On point queries
  + On range queries
  + On joins
  + On queries that use an Order By

However, a table can only have one clustered index

**Covered Queries**

* A covered query retrieves data from the index without accessing the table

select au Iname

from authors

where au Iname = 'Twain'

**Nonclustered Index Access**

* A nonclustered index is a balanced tree structure that provides random access to the data
  + It does not affect the order of the data in the table

select from authors

where au Iname = 'Twain'

**How to use index in optimization**

* Cluster index determines the actual physical storage order of the table data
* A table can only have one clustered index and more than one unclustered index.
* Three join types of join query between data tables
  + nested loop joins
  + merge joins
  + hash joins

**Transaction**

* A transaction is a logical unit of work
* Standard of Transaction (ACID)

Atomicity

Consistency

Isolation

Durability

* Transaction Syntax

begin transaction

commit transaction

rollback transaction

save transaction

**Lock**

* In a muti-user environment, transactions may access data simultaneously
* Locking is an automatic mechanism that isolate data to prevent conflicting modification
* Locking is dependent on transaction management

**Locking Schemes**

* **Allpages Lock**

It will lock data and index pages as they are accessed.

* **Datapages Lock**

It will lock data pages as they are accessed; index page are protected by latches.

* **Datarows Lock**

It will lock data rows as they are accessed; index pages are protected by latches

**Lock Types**

* **Shared locks**

It prevents others from channging data that is being read

* **Update locks**

It reserves data for modification by preventing others from modifying the data.

* **Exclusive locks**

It prevents others from accessing data that is being modified

**Transaction**

**Isolation level Definitions**

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Dirty Read** | **Nonrepeatable Reads** | **Phantom Reads** |
| **Level 0** | Allowed | Allowed | Allowed |
| **Level 1** | Prevented | Allowed | Allowed |
| **Leve l2** | Prevented | Prevented | Allowed |
| **Level 3** | Prevented | Prevented | Prevented |

* **Level 0:** read uncommit
* **Level 1**: read commit
* **Level 2:** repeatable read
* **Level 3:** serializable
* Now the default isolation level of Syase is level 1
* Now STECH default lock schema is page lock